Instituto Superior de Engenharia do Porto

Agile Modeling Data Warehouse Development

Frederico da Silva Couceiro

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Tecnologias do Conhecimento e Decisão

Orientador: Prof. Paulo Oliveira

Júri:

Presidente:

Doutor José António Reis Tavares

Vogais:

Doutora Maria de Fátima Coutinho Rodrigues

Doutor Paulo Jorge Machado Oliveira

Porto, [Outubro] [2012]

Acknowledgements

I wish to express my gratitude to all InovRetail team members, with distinct gratitude to Marco Soares, André Leão and Hugo Lopes for the support and opportunity to lead this project. Also to all the customers and their companies for their patience and contribute that helped define the product.

Special thanks to my tutor, Prof. Paulo Oliveira.

Abstract

Most of the traditional software and database development approaches tend to be serial, not evolutionary and certainly not agile, especially on data-oriented aspects. Most of the more commonly used methodologies are strict, meaning they're composed by several stages each with very specific associated tasks. A clear example is the Rational Unified Process (RUP), divided into Business Modeling, Requirements, Analysis & Design, Implementation, Testing and Deployment.

But what happens when the needs of a well design and structured plan, meet the reality of a small starting company that aims to build an entire user experience solution. Here resource control and time productivity is vital, requirements are in constant change, and so is the product itself.

In order to succeed in this environment a highly collaborative and evolutionary development approach is mandatory. The implications of constant changing requirements imply an iterative development process. Project focus is on Data Warehouse development and business modeling. This area is usually a tricky one. Business knowledge is part of the enterprise, how they work, their goals, what is relevant for analyses are internal business processes.

Throughout this document it will be explained why Agile Modeling development was chosen. How an iterative and evolutionary methodology, allowed for reasonable planning and documentation while permitting development flexibility, from idea to product. More importantly how it was applied on the development of a Retail Focused Data Warehouse. A productized Data Warehouse built on the knowledge of not one but several client needs. One that aims not just to store usual business areas but create an innovative sets of business metrics by joining them with store environment analysis, converting Business Intelligence into Actionable Business Intelligence.

Key-Words: Data Warehouse, Agile, Methodology, Modeling

Resumo

Tradicionalmente as metodologias associadas ao desenvolvimento de *software* e base de Dados tendem a ser sequenciais, não evolutivas e definitivamente não ágeis. As metodologias mais comuns tendem a ser rígidas, geralmente compostas por diversas fases ou níveis, onde para cada um destes existe um conjunto muito específico de tarefas a executar de acordo com uma ordem própria. Um exemplo claro é o *Rational Unified Process* (RUP), dividido em Modelação de Negócio, Requisitos, Análise e Desenho, Implementação, Testes e Disponibilização.

Mas como reagir quando as necessidades de um plano bem estruturado e desenhado se deparam com a realidade de uma empresa *start-up* com múltiplos projectos que pretende desenvolver um sistema de informação baseado em experiência de loja. Uma empresa com recursos limitados em que o retorno sobre a produtividade necessita de ser explorado ao máximo, algo que obriga a contínua procura de melhoria e subsequentemente a constante alteração de foco e requisitos.

Neste ambiente a necessidade de um desenvolvimento altamente colaborativo e progressivo é obrigatório. A constante alteração de foco implica a necessidade de um desenvolvimento iterativo, capaz de rapidamente reagir às alterações. Neste contexto o foco do projecto recai sobre o desenvolvimento de um armazém de dados e modelação de processo de negócio. Esta área específica é particularmente complexa, pois o conhecimento é intrínseco ao negócio. Como se trabalha? Quais os objectivos? O que é relevante para análise? Tudo isto é parte integrante da empresa e como tal conhecimento difícil de transmitir.

Ao longo do documento é pretendido explicar a razão pela escolha de uma metodologia Agile para o desenvolvimento. Como uma metodologia iterativa e gradual, permitiu desenvolvimento flexível com suficiente planeamento e documentação. Mais importante como foi aplicada de forma a desenvolver um Armazém de Dados com foco em Retalhe. Um capaz de responder as necessidades de vários clientes. Construído através do conhecimento de não um, mas múltiplos clientes através da análise dos seus requisitos. Este pretende armazenar não só a informação tradicionalmente associada a um Armazém de Dados de retalhe, mas sim juntar as áreas de um armazém de dados tradicional com informação recolhida do ambiente de loja. Desta forma criando um novo conjunto de métricas de negócio com o objectivo de melhorar o ambiente de loja de forma dinâmica.

Palavras-chave: Agile, Metodologia, Modelação, Armazém de Dados

Table of Contents

List of figures

List of tables

List of Acronyms

DW	Data Warehouse
BI	Business Intelligence
MDD	Model Driven Development
AMDD	Agile Model Driven Development
UML	Unified Modeling Language
RUP	Rational Unified Process
DB	Database
RAD	Rapid Application Development
XP	Extreme Programing
RDW	Retail Data Warehouse
IBM	International Business Machines
BA	Business Analytics
SPSS	Statistical Package for the Social Sciences
ORS	Oracle Reporting Services
ETL	Extract Transform and Load
DML	Data Manipulation Language
SLDC	Software Development Life Cycle
IAAS	Infrastructure as a Service
SAAS	Software as a Service
PAAS	Platform as a Service
CRUD	Create, Read, Update and Delete
TDD	Test Driven Development
KPI	Key Performance Indicator

1 Introduction

1.1 Motivation

InovRetail is retail Innovation Company founded in 2010, fully invested to Research & Development of new and advanced technological solutions designed to improve the customer retail experience, making Store Environments more appealing, dynamic and efficient, with measurable return to our clients.

This previous year the author was offered the opportunity to embark on a new project at InovRetail. Here the author was presented with the task of leading a small team responsible by the Database Area.

Nowadays data analysis is a must in supporting business strategies. On a data driven competitive market it is proven that information is a major resource, one vital to successful business. In this context Data Warehousing is must for running an enterprise of any size (Andrews Consulting Group, 2011). For the previous years the author has been working on Data Warehouse construction, customization and analysis for some of European major retailers. This thesis comes as an opportunity to apply the knowledge acquired during those years. More so, leading a team on the development of a Data Warehouse product with a fully innovative concept is an immense opportunity for personal and professional growth.

This development comes as a desired improvement to InovRetail software solution, the construction of a Data Warehouse, capable of satisfying Retail Customer's needs.

Developing a Data Warehouse with collaborative Dimensional Modeling means customer input. This project had the inputs of two major retailers Salsa and GNG, owners of most of Levi's franchise stores in Portugal.

1.2 Problem Background

1.2.1 Business

Retail Industry is a complex and highly demanding business. Customers are increasingly informed and demand impeccable service and differentiation. Retail is detail. InovRetail mission is to help Retailers improve Sales whilst optimizing their operating costs. Through the development of a technological platform designed to improve the customer experience in Retail Stores. The solution SEEPlus, integrates state of the art proprietary and best of breed partner's technology. InovRetail Store Atmospherics Predictive Models, Customer Profiling Solutions and Smart Layout Functionality will help create the best Retail experience. We are dynamic, so why shouldn't Stores be too?

Customer store experience is the intricate result of several different elements combined, all working together resulting in a decisive influence on the customer behavior translating into sales performance and operating costs of a store.

Below are shown some of the elements that interact and affect customer experience.

Figure 1 - Customer Experience Focus

Customer satisfaction and store productivity are not based on a single element but the combination of multiple. This brings several challenges like complex ecosystem, customer centric, interdisciplinary features, dynamic and variable, challenging to measure, unknown efficiency and improvement opportunities.

1.2.2 Technical

The approach taken to the technical aspects was greatly influenced by what InovRetail wanted to develop, and two main requisites, scalability and low initial investment.

Develop a full Software Suite with multiple modules on multi-environments, an application this size requires software capable of supporting:

- Team Development
- Integrated development environment
- Development Platform Support
- Architecture and Modeling Tools
- Task Collaboration
- Debugging and Diagnostics Tools
- Testing Tools
- Database Servers

Scalability is also a key requisite. The solution has to be easily scalable not only from a hardware point of view but also software. Hardware is a major point, mainly because InovRetail customer goal goes from small retail store to a TIER 1 business, also known as major retailer or wholesaler. The difference between these customer types is enormous. The main being investment, a small store can never compete with a TIER1 in term of investment or infrastructure. Needless to say they could both benefit from the product. An initial market analysis revealed that both business types were lacking any kind of support in this area. So a solution with low initial investment allows fighting in both ends. Offer small and stay small, or offer small and grow as much as desired.

Software scalability is also a concern, since the product is still in development, new customer mean new modules. New modules involve: new features, new architectures and new environments mobile, web or others.

The other major concern is low initial investment. Current economic situation is complicated. So finding customers willing to invest, especially on a completely new product from a start-up company is tough. In order to captivate customers it is necessary to gradually show, that the solution is not only a useful management tool but also a profitable investment. In order to do so, it is necessary to analyze customer's needs and with little investment possible deliver results.

As stated the main goal is to build a full software suite capable of supporting all these different business needs. The picture below shows an early base sketch of the main needed areas.

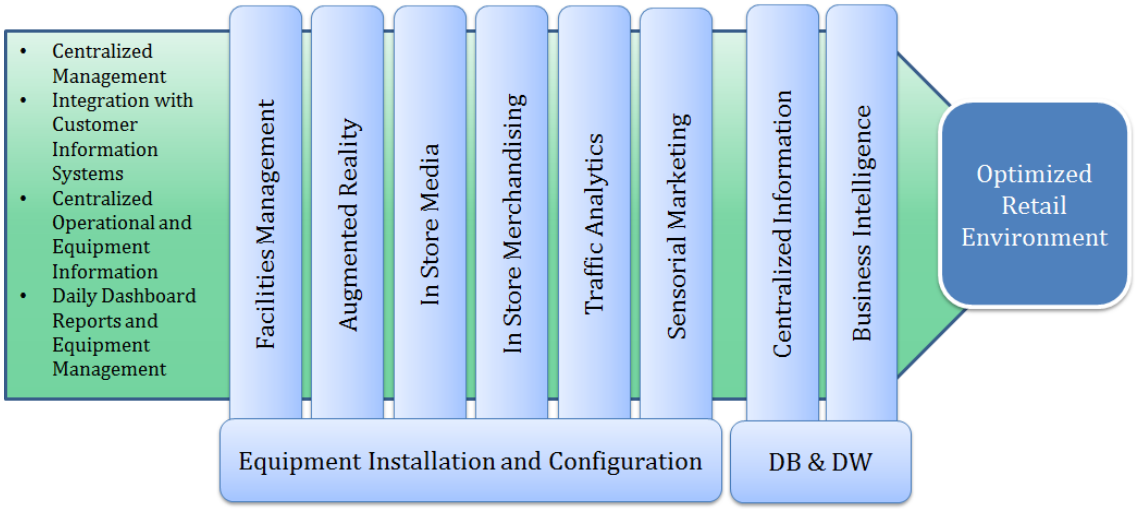

Figure 2 - Optimized Retail Environment Product Main Areas

Although software and hardware are a main problem, the focus is the construction of a Database architecture and posterior development. Primarily the construction of a Data Warehouse product capable of keeping, manage and aggregate all these different areas.

There are already Data Warehouse applications for retail in the market. The most commonly known from IBM and ORACLE, these solutions are impractical for most retailers, essentially because the investment required to implement a solution like this costs from several hundreds to millions, they take several month to implement and cover the retail business on a general level not the customer's specific business. Being so, what most retailer use is, either mixed Operational systems, basically developing the BI over the operational systems or they create a Data Warehouse on their own, making it a custom solution only applicable to the specific details of their business.

Developing a Data Warehouse capable of supporting different retail business is a difficult mission, largely because retail is a complex business. Depending on the business type, customer concerns change completely. For example if were talking about a company who owns franchise stores, they´ll be focused on Human Resources and Store environment. On the other hand a wholesaler will focus

on logistics and stock. Finally, the typical retailer focuses a little on all, from stock, product, sales, store management, warehouse management human resources, logistics to forecast. Each area is so big and specific that Oracle has at least one application for each one of these.

Project goal as previously stated is not to tackle all these areas. Some are unavoidable like sales, human resources and store or warehouse management. More importantly they are common to all customers. Most companies keep this information in Operational Information Systems either in integrated software solutions or custom ones. The solution intends to retrieve this information, join it with completely new areas like: atmospherics, traffic analytics, and energy. This is truly differentiating and allows the creation of a software solution capable of maximizing the retail environment and customer experience.

But, it also brings completely new problems:

- How to build and Data Warehouse without having an operational System?
- How to build a Data Warehouse capable of serving several customers with different business requirements?
- How to tackle Data Integration and Migration?
- How should the data be organized?
- What are the hierarchies, rollups or aggregations used with these dimensions?
- Is the information grain level the same for all customers?
- Interpreting Business Intelligence needs from multiple clients?

1.3 Objectives

The goal is to implement a product solution, able to improve customer experience in retail business, making them more appealing, dynamic and efficient with a measurable return for the client.

The objectives focused on 5 main points of improvement:

- **Traffic** - Improve customers shopping experience making the store appeal increase and visit numbers
- **Efficiency** -Increase customer purchase efficiency, by having more customers purchase with the same amount of visits
- **Loyalty** - Increase brand awareness on store
- **Energy efficiency** - Improved usage of energetic resources either by optimization or rationing energy usage. Reduce shop operational costs
- **Management** - Better business view through daily key performance indicator

Based on these objectives a group of factors was chosen to measure and affect. So that certain objectives could be attained, for each of these different business areas.

Marketing

- Sensorial marketing tools (Store smell and sight)
- Traffic analyses (People counters)
- Oriented marketing (Client specific campaigns)
- Heat maps
- Promotional campaign efficiency analysis
- Store dynamics
- Improve sales efficiency

Top management

- New management level apply macroeconomics concepts to micro management
- Innovations
- Brand tools
- Improved suppliers negotiation (New information sources)
- Improve sales efficiency

Commercial

- Improved Suppliers Negotiation (Commercial revenue)
- Traffic Analyses
- Promotional Campaign efficiency analysis
- Improve promotional campaigns

Operations/Facilities

- Continued store environment motorization
- Improved store resource allocation
- Equipment centralized management
- Operational cost reduction

1.4 Document Structure

This thesis is divided in five main chapters. The current chapter, Introduction, presents the main motives and a preview of the project objectives.

This is followed by State of Art divided in four main topics DW/BI Solutions, Development Methodologies, Development Tools and Hardware Solutions. This chapter intends to show a general view of the existing options for each of the essential main areas on project development.

The third chapter Agile Modeling focuses on the chosen development methodology, with special emphasis on its application to Data Warehouse construction and why it was the chosen methodology.

The fourth chapter Solution describes the implemented solution on its different levels, from defined Milestones to developed modules.

Finally the fifth chapter Conclusion conveys the acquired concepts during project development and future work.

2 State of Art

2.1 DW/BI Solutions

2.1.1 Data Warehouse Solutions

A data warehouse is a database that stores data from many business areas in the company, in a way that allows this information to be easily combined for reports or analysis tools.

Ralph Kimball

"A data warehouse is a copy of transaction data specifically structured for query and analysis." (Kimball, 1996)

Bill Inmon

"A data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process." (Inmon, 1992)

Like any other system data has to be organized or structured. Most Data Warehouses divide their information between Data Marts. A Data Mart is a set of data specific to one given subject or functional area, like sales or stocks. A Data Warehouse is usually comprised of several Data Marts.

There are not many Data Warehouse solutions for retail in the market. The size and complexity of a retail business make it almost impossible for a new software company to create a new product that can satisfy the business needs of a company like SONAE, AHOLD or TESCO.

A Data Warehouse is not a software solution is a business one, and retail is a particular difficult business. Let´s look at some facts:

- Retail is country oriented

- Portugal taxes are different from UK or Spain and therefore accounting needs vary
- Seasons in Portugal are different from Brazil, so marketing needs differ
- Even operational costs are different impacting net sale
- Some businesses are also wholesales, so supplier and client needs differ
- Some use multiple small stores other bet in big stores, so stocking and store front have different needs.
- Some businesses have their own production others do not
- Some businesses are franchised
- Some have their own warehouse and logistics companies others buy it as a service

These are just some of the major differences you can find when looking at different retail business, and has stated a Data Warehouse is a Business Solution.

The reason is simple companies develop or buy Data Warehouse solution in order to improve Business and Informational Reporting, they want a better view, more control or even a forecast of their business and everyone is different.

That being said there are still solutions in the market, namely three:

- Oracle´s Retail Data Warehouse (RDW)
- IBM Retail Data Warehouse a component of IBM's Retail Business Intelligence Solution
- SAP Business Information Warehouse

IBM Retail Data Warehouse

IBM is a successful brand in Systems Information with clients going from Estée Lauder to Visa Inc. The company has a success record a mile long and a tradition of success in Data Warehouse implementation. IBM solution is comprised of a comprehensive data warehouse and Data Mart accelerators that contain data warehouse design models, business terminology model and business solution templates.

“IBM Retail Data Warehouse models accelerate the development of business intelligence reporting and analysis solutions for the retail and wholesale industries. They enable the consolidation of trusted information in support of organizational initiatives.” (IBM, 2012)

Major Product assets:

- Delivers competitive advantage by enabling the consolidation of clean, meaningful retail data across multiple channels and products
- Supports key retail requirements including customer management, merchandising management, products & services management, store operations management, corporate finance management, supply chain, multi-channel, regulatory compliance visibility and more
- Accelerates both business requirements gathering and supporting data model design
- Provides a solid basis for statutory reporting and relationship management supporting decision support, real-time analytics and predictive modeling
- Proven retail industry specific models minimize development costs and reduce project risk. Incremental approach supports rapid delivery of business functionality

Oracle RDW

Much like its competitor, Oracle RDW is a known product with already thirteen versions and a long history of successful implementations. Oracle RDW has some of the biggest retailer in the world has clients. This product was specifically built for retail and supports most of if not all the retail business areas to some extent.

"Oracle Retail Data Warehouse (RDW) provides retailers with a fully functioning out-of-the-box solution, complete with Oracle Retail Business Intelligence. This serves as a foundation on which retailers may customize and extend the data model and reporting as they see fit." (Oracle, 2008)

Oracle base system supports over 700 retail-specific facts and dimensions support over 200 metrics / KPI's, spanning the following functional areas:

- Accommodations
- Allocation
- Balance of contract
- Beginning and ending stock values
- Business metrics
- Commitment total units and value
- Comp
- Comp store
- Customer loyalty
- Discount coupons and scanned Items
- For reports

- Gross margin return per dollar of investment (GMROI)
- In store markdown
- In transit
- Loss prevention voucher
- Markdown
- Media
- Net cost
- On order
- Organization, sales value by type
- Overrides
- Pack sales
- Plan inventory
- Plan mark-ups and mark-downs
- Plan sales & profit
- Price
- Profit per unit of allocated space
- Promotions
- Reserved
- Return to vendor
- Returns & replacements
- Sales & gross margin
- Sales & profit
- Sales and returns by tender type
- Sales per unit of allocated space
- Stock adjustments
- Stock ledger receipts
- Stock movement receipts
- Stock on hand
- Stock turn retail value
- Store traffic
- Supplier contracts & availability
- System metrics
- Transfer

SAP Business Information Warehouse

“It was originally named SAP BIW (Business Information Warehouse), then abbreviated to SAP BW, but is now known as "SAP BI" at the end user level. In contrast, "BW" is still used to describe the underlying Data Warehouse Area and Accelerator components. It is often used by companies who run their business on SAP's operational systems.” (Wikipedia, 2012)

SAP NetWeaver Business Warehouse (SAP NetWeaver BW) is an analytical, reporting and Data Warehousing solution developed by SAP AG. SAP solutions is not retail specific, but supports decision making in the most important areas like:

- Price and promotion optimization
- Demand forecasting
- Product affinity metrics analysis
- Loss prevention
- Merchandise and assortment planning
- Point of sale analysis

The following graphic shows how the SAP BW concept is structured. Data Warehousing, BI Platform and BI Suite represent the core areas of SAP BW. (SAP, 2012)

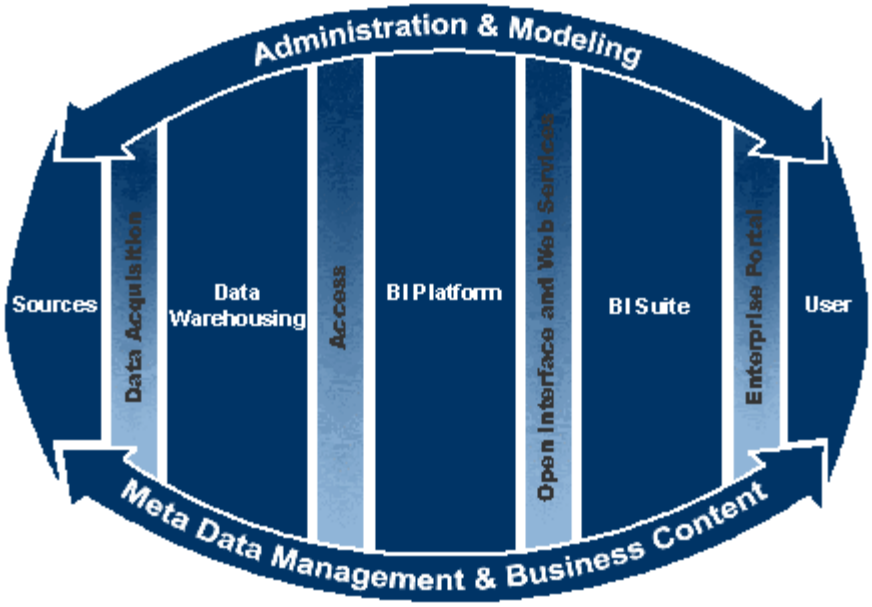

Figure 3 - SAP BW Solution Architecture (SAP, 2012)

Product Comparison

After a careful analysis these solutions cover approximately the same areas. They both strongly recommend that the Data Warehouse solutions be a part of a retail suite mainly the operational product in Oracle´s case Retail Merchandizing System (RMS) for IBM the Retail Business Intelligence Solution and SAP NetWeaver. All have very large implementation cost, which vary from performance, scalability and storage capacity and number of purchased modules.

- With the exception of SAP solution, both recommend to be installed on their own Hardware solutions, IBM Smart Analytics System and Oracles Exadata.
- The implementation of any of these solutions, takes considerable time
- Any of these implementations needs considerable resources
- These solutions are particularly expensive
- Normally require customization in order to fit all the business needs
- Neither solution is aimed at small retail customers actually their not even aimed for medium sized ones

“Oracle’s Exadata server smaller installation is nothing less than 21 Terabyte’s, IBM system works on a different base, the server is divided in modules but still the smaller installation ranges the hundreds of Gigabytes.” (Howard, Philip, 2011)

So what other solutions are there?

Complete Data Warehouse solution for retail not many more, there are other products usually associated with operational or POS systems that offer Data Warehouse features, but not true Data Warehouse Systems. For most small retailers this is more than fine:

- Most of their needs can be satisfied with small alterations to the operational system
- Insufficient data volumes to create real performance problems
- Insufficient or inexperienced resources to pursue complex KPI analysis or Business Intelligence
- Focus on operational activities
- The truth to most retailer is they don´t know or have the resources for running the business and managing a full Data Warehouse implementation. Some do not even truly grasp what a Data Warehouse is its advantages to business.
- That being said, there are those who want true Data Warehouse Systems. Those usually outsource, hire or through internal development create their own custom solutions. This is true for most small and medium retailer but also major ones like one of biggest Portuguese Retailer

The table below intends to offer a better comparison view between existing data warehouse solutions.

Table 1 - Data Warehouse Solution Comparison

Parameters	Data Warehouse Product Solution	Operational System Based	Custom Data Warehouse
Cost	High	Depend on needed customization, usually low investment needed	Depend on project size, can be controlled trough Agile development and prototyping
Business Areas	Covers Most retail Business Areas	Few can only cover what the operational or POS system has available.	Every possible source of information from operational systems, to flat files to manual insertion
Employee Involvement	High, this kind of project usually has high impacts on how each area works. And involves a great part of the company	Low, only the end users that will be affected by the new features.	Flexible depending on the number of business areas involved or the size of the project
Scalability	Medium. Customization is viable but usually with associated costs on development and Hardware	Low. Difficult to create a full Data Warehouse solution from existing system	Cost associated with development and Hardware but independent from solution. Added supplier choice
Licenses	Depend on signed contract. Usually with a high cost	Usually associated with a number of users	None

Each solution has its own advantages and shortcomings. More importantly depending on the current business status any of these solutions can be the right one to choose from.

2.1.2 Business Intelligence

BI technologies provide historical, current and predictive views of business operations. Common functions of business intelligence technologies are reporting, processing, analytics, data mining, process mining, complex event processing, business performance management, benchmarking, text mining, predictive analytics and prescriptive analytics.

“Business intelligence (BI) is defined as the ability for an organization to take all its capabilities and convert them into knowledge. This produces large amounts of information which can lead to the development of new opportunities for the organization. When these opportunities have been identified and a strategy has been effectively implemented, they can provide an organization with a competitive advantage in the market, and stability in the long run (within its industry).” (Rud, 2009)

"Business intelligence aims to support better business decision-making. Thus a BI system can be called a decision support system (DSS)." (Power, 2007)

There are several Business Intelligence Products currently on the market. Actually there are so many it is probably impossible to cover them all. Most require or strongly recommend a Data Warehouse System in order to access information. Others require only an information source be it a DW or an Operational System.

Below are some of the more commonly known:

- Oracle Enterprise BI Server 11g1
- SAP Business Objects Enterprise XI r4
- Microsoft Integrated BI offering 2008/2010
- Information Builders WebFOCUS 8
- QlikView 11
- SAP NetWeaver BI 7.3
- Style Intelligence 11
- Pentaho BI Suite 4
- IBM Cognos Series 10.1
- SAS Enterprise BI Server 9.2
- Board Management IntelligenceToolkit 7.1
- Microstrategy 9
- Oracle Hyperion System 9
- BizzScore Suite 7.3
- JasperSoft 4.5
- Tableau Software 6.1
- Microsoft Reporting Services
- SAP CRYSTAL REPORTS

- Oracle Reporting Services 11g
- IBM Statistical Package for the Social Sciences (SPSS)

Not all will be covered, only the five more commonly used in retail business.

Oracle Business Intelligence Enterprise Edition (OBIEE), the newest and most commonly used in recent year, its known for a fact that two of Europe's major retailers are currently transitioning part of their BI structures to OBIEE.

Oracle Reporting Services an integrated solution that comes with Oracle Retail Suite.

Business Analytics an IBM solution resulted from a merge between an independent solution COGNOS bought by IBM in 2008 and SPSS in 2010.

Microsoft Integrated BI a different kind of business intelligence, divided in three main products.

Microstrategy 9 a product from Microstrategy, Inc. a company founded in 1989, with Business Intelligence in mind.

OBIEE

The Oracle Business Intelligence Suite Enterprise Edition Plus (EE) is a comprehensive suite of enterprise BI products, including:

- Interactive dashboards
- Full ad hoc querying
- Proactive intelligence and alerts
- Enterprise and financial reporting
- Real-time predictive intelligence
- Disconnected analytics

"Oracle Business Intelligence Suite EE Plus platform is based on a proven, modern Web Services-Oriented Architecture that delivers true next-generation BI capabilities." (Oracle, 2012)

2.1 DW/BI Solutions

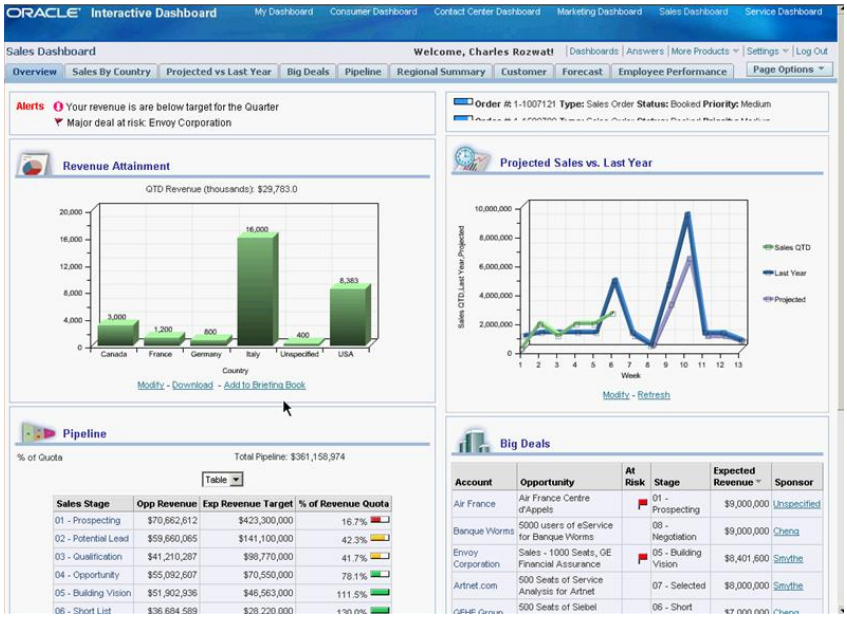

Figure 4 - OBIEE Dashboard Overview (Oracle, 2012)

OBIEE implies the installation of Oracle BI Server a true BI server that is designed to be:

- Highly scalable
- Optimizing concurrency
- Parallelism
- Centralized data access and calculation
- Designer for multiple user access anywhere in the enterprise

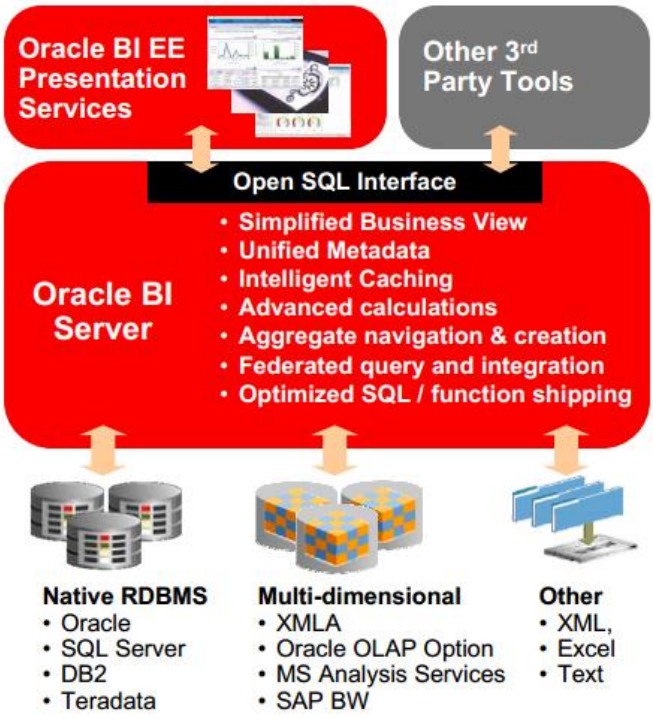

Figure 5 - OBIEE Architecture (Oracle, 2012)

Basically Oracle Bi is a high-impact query, reporting and analysis server based on three information layers.

The Physical layer describes the physical data implementation where the data sources, the tables and columns, the relationships, connections parameters and so on are defined.

The logical data model layer is first abstraction layer and its aim is to design logically a dimensional data (business model) model with fact, dimension, hierarchy and calculations from the physical layer elements (the mapping part of the name).

The presentation layer is the third and last abstraction layer and is in charge of the presentation of the table and columns of the logical data model to the user and client application.

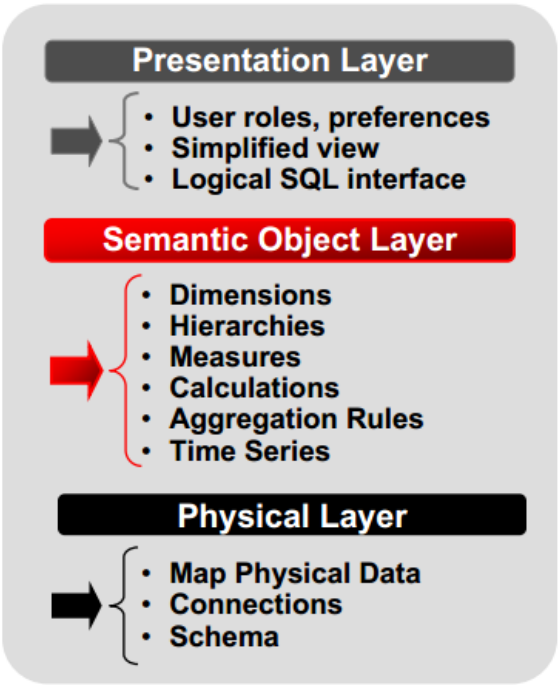

Figure 6 - Oracle Enterprise Information Model (Oracle, 2012)

The presentation layer feeds OBIEE Presentation Services and other components mainly Oracle Interactive Dashboards, Oracle Answers, Oracle Delivers and Oracle BI Publisher.

Table 2 - OBIEE Delivery Components

Delivery Components	Objective
Oracle Interactive Dashboards	Live, intuitive, personalized dashboards • Immersive experience with limitless dashboard to dashboard navigation • Dynamic, event-driven guided analytics lead users from insight to action
Oracle Answers	Powerful ad-hoc analysis via 100% Web UI • Create & modify charts, pivots and dashboards • True business user self-sufficiency via intuitive, simplified business model
Oracle Delivers	• Automated problem / opportunity detection and alerts • Notification and content delivered to any device in multiple formats (PDF reports, Briefing Books, etc.) • Multi-step analysis, analytic workflows, and business process integration
Oracle BI Publisher	• True "pixel-perfect" reporting and publishing • Report layout using familiar tools like Microsoft Word and Adobe Acrobat • Single product for all document needs – invoices, checks, financial statements, government forms, etc.

Oracle Reporting Services

Oracle Reporting Services was somewhat Oracles OBIEE predecessor. ORS is a separated product in its own right, one that still has many more clients than OBIEE.

"Oracle Reports a component of Oracle Fusion Middleware is Oracle's award-winning, high-fidelity enterprise reporting tool. It enables businesses to provide instant access to information to all levels within and outside of the organization in a scalable and secure environment. Oracle Reports consists of Oracle Reports Developer - a powerful, declarative WYSIWYG reports design tool and a J2EE 5.0 based Oracle Reports Server with multi-tier architecture to access any data sources, generate reports in any popular format for web and paper, and to burst and distribute reports to any destination. Oracle remains committed to the development of this technology, and to the ongoing release as a component of the Oracle Fusion Middleware platform." (Oracle, 2012)

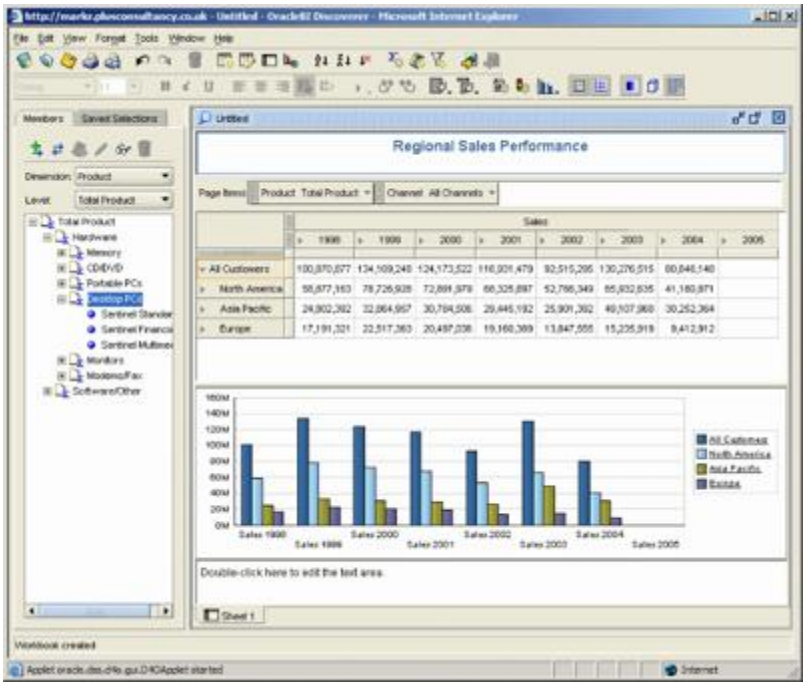

Figure 7 - Oracle Reporting Services

Oracle Reports Services provides a scalable, flexible architecture for the distribution and automated management of report generation engines on the same server and across multiple servers. Additionally, it caches report output for reuse on similar requests. It integrates into standard Web environments with JSPs, Java servlets, and Web Services. It enables you to run reports on both local and remote application servers and to implement multi-tiered architecture for running reports.

When used in conjunction with JSPs, Java servlets, or Web Services, Oracle Reports Services enables to run reports on any platform from a Web browser using a standard URL syntax.

This BI Tool is still utilized worldwide, mainly because Oracle offers this solution in association with their Retail Suite. But as can be quickly realized the development of this solution has stopped and is slowly being replaced by the new Oracle BI Product OBIEE.

IBM Business Analytics (Cognos)

The product is a complete Business Intelligence application capable of supporting virtually all BI requisites a modern retail company can imagine. Much like its competitor OBIEE it offers a complete package from data integration to a complete set of information managing and visualization tool like Reports, Dashboards and Analysis like trend Analysis and What-if.

“Cognos (Cognos Incorporated) was an Ottawa, Ontario-based company making business intelligence (BI) and performance management (PM) software. Founded in 1969, at its peak Cognos employed almost 3,500 people and served more than 23,000 customers in over 135 countries.

Figure 8 - Cognos Dashboard

In January 2010, as part of a reorganization of IBM Software Group, Cognos software and software from recently acquired SPSS were brought together to create the Business Analytics division." (Wikipedia, 2012)

Things You Should Know About Cognos (EMA, 2010)

- Delivers features and solutions for business intelligence, predictive and advanced analytics, financial, performance and strategy management, governance and compliance, and analytic applications
- Built-in collaboration and social networking, ability to create decision communities, produce decision templates and capture lineage of decisions throughout the system
- Major mobile compatibility
- The platform enables full interactivity with reports including drill-down, scheduling, etc.
- SPSS Modeler and statistics are integrated into the platform providing predictive analysis and insights
- Designed to make information more actionable
- includes lifecycle management tools
- Can be delivered to most any sized client with a well-defined growth path

COGNOS has the ability to create predictive analytics with a higher emphasis on mathematical and forecasting tools like SPSS. From all the products this seems to be the most advanced in terms of modern BI requisites. Nowadays companies do not just want to see the information in their systems. They want to predict pattern profile client and create group clustering and segmentation. It is clear by IBM´s roadmap that this is no coincidence.

“In the past 5 years, IBM has invested US$ 14 billion in the acquisition of 24 software companies. With Cognos being one of them, the overall offering starts to clearly benefit from some of these new technologies. Following the consolidation in the Business Intelligence market a few years back, when IBM acquired Cognos and SAP acquired Business Objects, Cognos clearly seems to digest well the transition into the IBM family” (Vanleenhove, 2011)

Microsoft Integrated BI

Microsoft integrated BI tools aim to be familiar, integrated and enterprise-ready. Microsoft Business Intelligence proposes to quickly access the timely, relevant, and accurate information that drives better decision making and impacts the bottom line. Microsoft bets on already existing trusted and familiar tools in order to solve business needs of companies of all sizes. (itelligence, 2010)

“Each is a powerful tool and a market front-runner in their own area. Microsoft Office is probably the most common application of them all. Enough to say it’s currently used by more than 750 million users worldwide according to Microsoft.” (Warren, 2010)

Microsoft solution is the most economical solution off them all. Microsoft suite is based on three applications that are common knowledge to any IT graduate in the world.

- Office
- SQL Server 2008 R2
- SharePoint 2020

“Microsoft SharePoint 2010 makes it easier for people to work together. Using SharePoint 2010, your people can set up Web sites to share information with others, manage documents from start to finish, and publish reports to help everyone make better decisions” (Microsoft, 2012).

Figure 9 - What is SharePoint (Microsoft, 2012)

SQL Server is used trough the world with clients that range from one man company's to the largest corporations on earth, it's the better documented and the easiest to find solutions for due to the wide range of forums, tutorials, workshops and certificates that help users become proficient in SQL Server.

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet). (Wikipedia, 2012)

The worth of the applications is unquestionable but their real value has a BI Source comes from how well they work together and the way they complete each other.

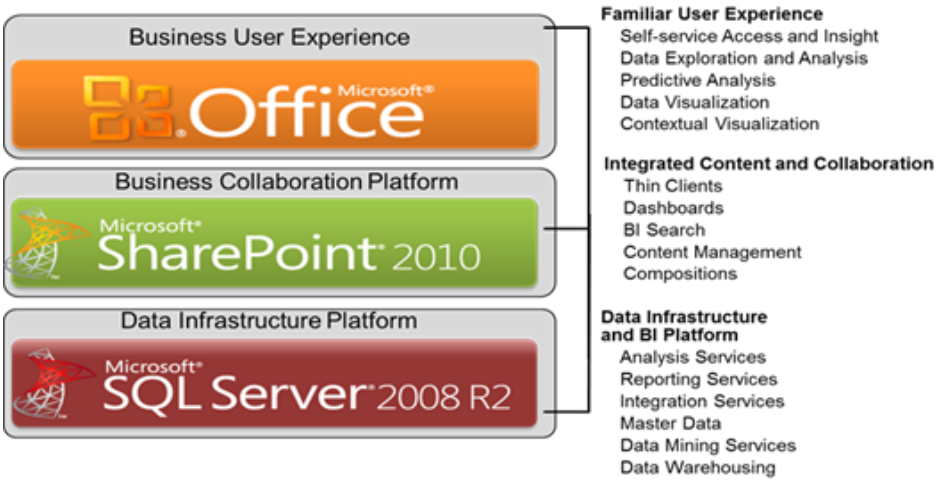

Figure 10 - Microsoft Tools Advantages (itelligence, 2010)

SQL Server is used as an architecture base, allowing data storage and integration trough Microsoft Integration Services, SharePoint allows web availability, user and role control and a user interface system capable of distribution Excel Sheets, Dashboards, PowerPivots and many more data visualization tools available not only in office but also in Reporting and Analysis Services.

Microstrategy 9

The MicroStrategy platform makes Business Intelligence faster, easier, and user-friendly. MicroStrategy offers a high performance, scalable enterprise Business Intelligence platform capable of delivering BI to large user populations through web browsers, mobile devices and Office applications. (MicroStrategy Inc., 2012)

MicroStrategy Solution aims to enhance productivity, monitor trends, discover anomalies and forecast business opportunities.

Like most of its competitors MicroStrategy offers a wide feature range like, scorecards and dashboards, advanced and predictive analytics, closed-Loop BI, high performance BI and reports.

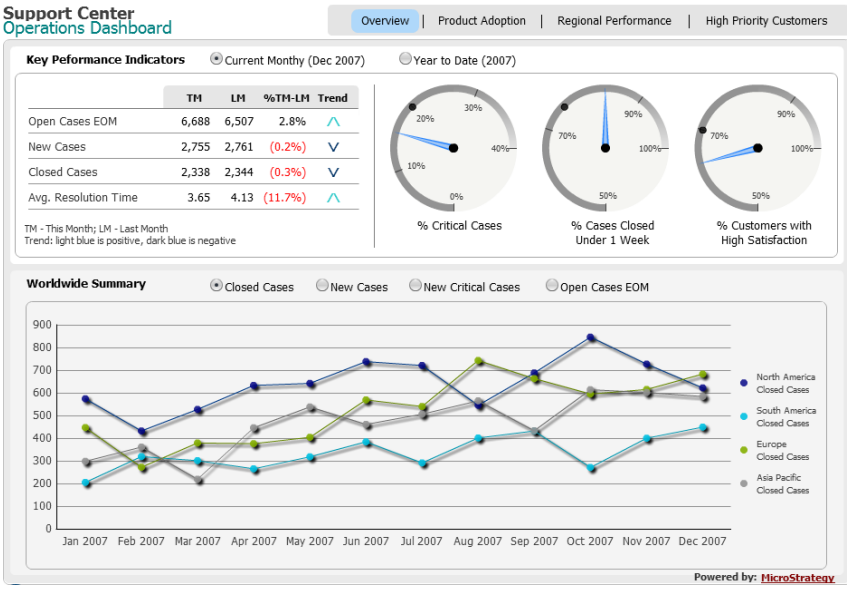

Figure 11 - Microstrategy Dashboard View

Microstrategy bases its structure in 8 main points (MicroStrategy Inc., 2012):

- **Secure** - Multi-dimensional security model
- **Scalable** - Scalable Business Intelligence
- **Easy to Maintain** - Suite of products that help reduce maintenance.
- **Easy to Use** - Top performer in multiple "Ease of Use" categories on Gartner Study
- **Single Unified Architecture** - Unified, organically-developed enterprise class Business Intelligence technology
- **Single, Reusable Metadata** - Object-oriented metadata that defines the enterprise's business layer in a single repository
- **Pervasive** - Allows users to access, monitor and analyze their information on mobile devices, web browsers, or Microsoft Office applications
- **Real-Time WYSIWYG Report Design** - Highly formatted reports are built using what-you-see-is-what-you-get

Product Comparison

Comparing these solutions against each other is a hard endeavor, one that even produced specialized companies like Business Intelligence Tool Box a company specialized in comparing just the solutions against each other.

Choosing the right solution for a specific company and implementing it is an entire different subject, excluding the supplier company's there are thousands of company's offering BI consultancy and product implementation services.

These are very complex product with lengthy learning curves, not just from a technical perspective actually the technical part might even be the less important, the business knowhow required to fully understand these products capability is astonishing and a must in order to take advantage of all its capacities.

There are two companies that currently stand out on BI world, IBM and Oracle, both product have incredible feature lists with minor differences from each other. BI is a complex market for example 5 years ago Microstrategy was market leader and the only tool to offers what is now common like Unified BI Architectures, Dynamic Enterprise Dashboards. Nowadays Oracle and IBM became BI monsters, currently offering the most complete BI products. More than just the product these companies distanced themselves, investing not just in having the most modern and capable product but more importantly they developed a complete strategy of integration, support, maintenance and business consultancy so that their clients can get the most return out of these solutions. These features come with a price and a steep one, most OBIEE and COGNOS solutions are million dollar projects, not for every retailer in the world.

Microstrategy and Microsoft still play in the big leagues, MicroStrategy's customers include eBay, Facebook, GUESS?, Inc., McDonald's, Whole Foods, Banco Bilbao Vizcaya Argentaria, SWIFT,Edmunds.com, and LinkedIn and Microsoft like Siemens, Goodyear, Dunlop, Caterpillar and many more. The main difference between these products and IBM and Oracle is the capability of serving small business, MicroStrategy's has free base licensing and Microsoft offers amazing partnership deals not to mention that many companies have already several of the necessary product set up.

2.2 Development Methodologies

A software development methodology or system development methodology in software engineering is a framework that is used to structure, plan and control the process of developing an information system (Wikipedia).

"There are currently several software development methodologies, the following are among the most common" (Bednar & Robertson, 2006):

- **Waterfall** - Linear framework type
- **Prototyping -** Iterative framework type
- **Incremental** - Combination of linear and iterative framework type
- **Spiral** - Combination linear and iterative framework type
- **Rapid Application Development (RAD) -** Iterative framework type
- **Rational Unified Process (RUP)** - Iterative software development
- **Extreme Programing (XP) -** Agile software development
- **Agile -** Iterative and incremental development

This methodology research is focused on finding the best suited for the project at hand. Due to the amount of existing development methods, the author focused not on describing each one but instead categorize them in common groups.

All methodologies are controversial. Depending on the articles or comments read, there is never a consensus on which methodology to use, even when the same project is in question.

So what makes it so hard to choose the correct methodologies?

The main thing to remember is that developing methods are tools. A resource used to organize, structure, plan work and has any tool people grow accustomed to it. Some people do not even realize they are using a development methodology, they are taught to work a certain way and it becomes a habit, something so natural and ordinary that modifying it completely sabotages their work.

It is a known fact, people are adverse to change, and has such specific opinions are biased, what research allowed to clearly find, was that methodologies tend to diversify between too extreme poles "Cowboy hacking and Micromanagement" (Boehm, 2002), from unplanned or no development methodology to inch pebble planning where every task needs validation and sign off.

Table 3 - Cowboy hacking versus Micromanagement

Figure 12 - Methodologies Poles

• No Planning	• Micromanagement
• No bureaucracy	• Step by Step Approvals
• Immediate Development	• Gather Requirement
• Programmer Autonomy	• Employee Specific Tasks
• No Development/Design Framework	• Strict Development /Design Framework
• Independent Developers	• Companies
• Loose Requirement Definition	• Strict contract binding Requirements
• Small Project	• Big Projects

This division clearly shows why a big project cannot be managed in a cowboy style and a small one in a Micromanaging way. A small project does not have the capacity or resources to have different teams for each phase. Also there is no need to contract bind every single defined requirement. On the other side a big project needs to have development/design frameworks the lack of such measures could mean that every developer worked on it preferable framework making it impossible to join the final work. A loose requirement definition could mean disaster on a contract binding project, if what you pay is not what you get.

Besides these two opposites, methodologies can also be divided according to the way they work. They are usually Linear, Interactive, Agile or Linear and Interactive.

Linear Methodologies follow a defined path, Waterfall is a perfect example. First business requirements, then design then Implementation and Integration, finally validation and maintenance. Linear methodologies are just that linear they follow a strait path from begin to end.

“The first formal description of the waterfall model is often cited as a 1970 article by Winston W. Royce” (Wikipedia, 2012).

This methodology has was defined in 1970 is clearly an outdated concept, the most criticized and the main reason for its nom usage is the fact that for any non-trivial project is impossible to completely close a phase in order to step to the next one. Although impossible might be an exaggeration, the concept is simple. Adjustments during a product lifecycle are a normal occurrence. In a phase like requisite gathering is clear. At this stage either if the customer did not understood what was supposed to be defined or there were requisites that were only discovered while implementing. The main fact remains, backtracking to the start of a previous phase and start again on all the sign-off procedures, requires too much time and effort and delays all future phases.

The evolution to iterative models and mixed linear/interactive models is an understandable step. "Iterative Development is an approach to building software in which the overall lifecycle is composed of several iterations in sequence. Where each iteration is a self-contained mini-project." (Larman, 2004)

"The Unified Rational Process (RUP) trademarks of IBM, and first published in The Unified Software Development Process (ISBN 0-201-57169-2) and published in 1999 by Ivar Jacobson, Grady Booch and James Rumbaugh" (Wikipedia, 2012). RUP is a clear advancement on an iterative and incremental software development process framework.

This methodology is based on four simple principles:

- Iterative and Incremental - "Phases are divided into a series of time boxed iterations. Each iteration results in an increment, which is a release of the system that contains added or improved functionality compared with the previous release. Although most iterations will include work in most of the process disciplines (e.g. requirements, design, implementation, testing) the relative effort and emphasis will change over the course of the project." (Wikipedia, 2012). The picture below reflects this concept to perfection in each phase there is an increased focus on a specific task. Nonetheless there is still activity on other tasks generally on the next or previous ones

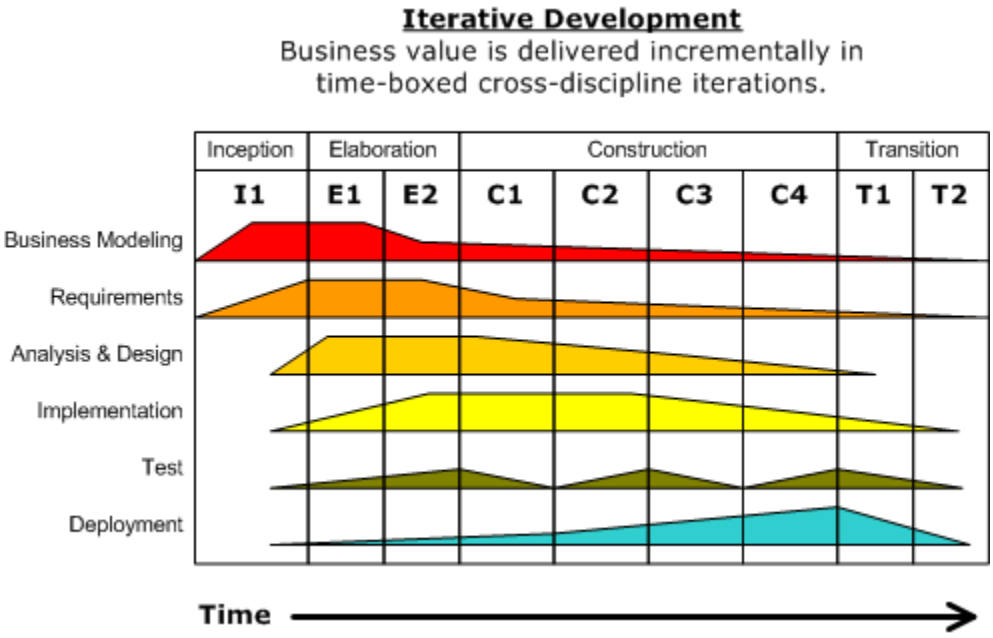

Figure 13 - Iterative Development

- **Use Case Driven** - Examples are the most proven and simple way of getting a point across, case driven methodologies applies that, design and describe cases that explain scenarios, be it to capture functional requirements or define iteration content. Every iteration takes a set of use cases or scenarios from requirements to implementation, test and deployment

- **Architecture Centric** - In order to correctly build anything a plan must exist. Instead of focusing on your every single need, focus on what you can control model-driven scenarios that describe what the system does and address the functional requirements. The Unified Process insists that architecture sit at the heart of the project team's efforts to shape the system

- **Risk Focused** - Simply put, tackle the big problems early at project life cycle. There is less risk and big architectural alterations have less impact. Furthermore it allows for better plan after these barriers are overcome

These simple principles extend into four phases:

- **Inception** - Scope the system effectively as a foundation for validating initial cost and budget

- **Elaboration** - Produces basic architecture; plan of construction significant risks identified; major risks addressed and majority of the system requirements

- **Construction** - Produces beta-release system. Largest phase in the project. In this phase the remainder of the system is built on the foundation laid in elaboration

- **Transition** - Introduces system to users, basically deployment of solution

"While the Iterative Model solves many of the problems associated with the Waterfall Model, it does present new challenges." (Intellitech Consulting Inc., 1997)

- The user community needs to be actively involved throughout the project. While this involvement is a positive for the project, it is demanding on the time of the staff and can add project delay
- Communication and coordination skills take center stage in project development
- Informal requests for improvement after each phase may lead to confusion - a controlled mechanism for handling substantive requests needs to be developed
- The iterative model can lead to "scope creep," since user feedback following each phase may lead to increased customer demands. As users see the system develop, they may realize the potential of other system capabilities which would enhance their work

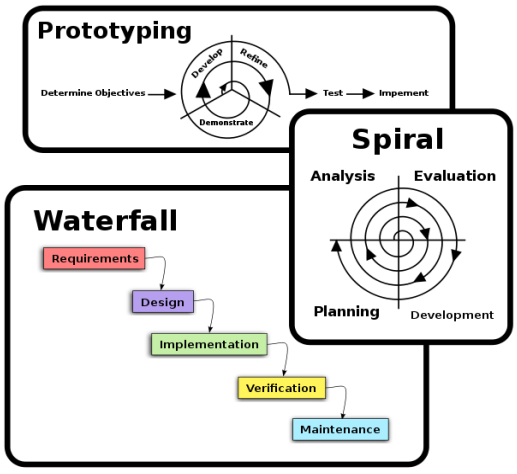

Figure 14 - Traditional Development Methodologies

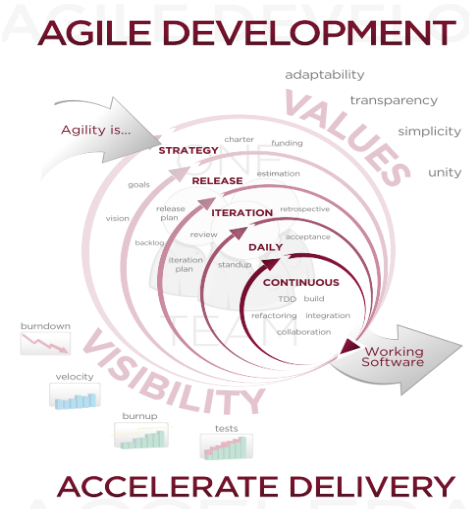

Figure 15 - Agile Development Methodologies

Agile Methodologies are not so much an evolution from iterative but more of a different approach. Agile Software Development is a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and encourages rapid and flexible response to change. It is a conceptual framework that promotes foreseen interactions throughout the development cycle. The Agile Manifesto (Beck, 2010) introduced the term in 2001.

Agile project management is guided by these ideologies and principles:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Twelve principles underlie the Agile Manifesto (Beck, 2010), including:

- Customer satisfaction by rapid delivery of useful software
- Welcome changing requirements, even late in development
- Working software is delivered frequently (weeks rather than months)
- Working software is the principal measure of progress
- Sustainable development, able to maintain a constant pace
- Close, daily co-operation between business people and developers
- Face-to-face conversation is the best form of communication (co-location)
- Projects are built around motivated individuals, who should be trusted
- Continuous attention to technical excellence and good design
- Simplicity- the art of maximizing the amount of work not done - is essential
- Self-organizing teams
- Regular adaptation to changing circumstances

2.3 Software Tools and Hosting Services

Choosing the correct tools for the job is probably one of the most important decisions to take on a software development project.

Provide the right tools. A project needs the appropriate tools that facilitate productivity for the team. Tools include hardware as well as design, programming and test productivity aids. “The cost

justification of these tools is relatively easy. Training in the new tools or techniques is also essential to ensure that they are used to their full advantage.” (Perks, 2003).

The statement above from an IBM developer shows that choosing a tool is more than simply using an application to program on a preferable coding language. A tool must be comprehensible, useful, functional and most important, reliable. Choosing the wrong one for the job at a startup company can mean months off work to waste and a disaster that can cost the company its sustainability.

There are two main groups on software developing tools free and paid, both with advantages and disadvantages. Freeware tools have a clear advantage: their price. For startup companies this is evidently a win. This does not automatically mean they are worse than paid tools, but there is usually a tradeoff involved. Free tools rely on communities, groups of people that create and support the tool through its lifecycle. The problem with communities is simple, they create and support the tool as long has it interests them. Also they are not paid thereof people can demand only request and if the request interest enough people maybe it will be a new version or feature.

Paid tools on the other hand are commonly expensive, especially good ones. They need licensing they define what you can and cannot do with their product and infringement of these rules can mean legal lawsuits. That being said, when paying for a product the consumer can demand. Company wise this is very important if the product does not work you can complain and request support. A clear example is Microsoft Internet Explorer Web Browser, it is common knowledge that there are better Web Browsers in the market like Chrome or Firebox, but the simple fact that none of those, have a company that ensures support for most versions of their product and not just the last one, makes web business products still choose IE has the targeted browser to use.

Accountability is a major factor in business and the bigger the business the more important it becomes.

Table 4 - Free versus Paid Development Tools

Development Tool Kits	Free	Paid
Price	Free	Depends on tool, good may be expensive
Support	Communities and Forums	Guaranteed by the supplier
Language	Reduced	Can vary, but normally covers a wide range
Reliability	Depends, some have very strong development teams like HTML5 other practically nonexistent	Guaranteed by the supplier
Training	Depends usually harder to get than paid, although there are cases were external companies offer training	Thousands of tutorials, certificates and professionals. Easy access to training trough the world
Lifecycle	Depends on success and existing community	Guaranteed by supplier. Sometimes even assured Retro compatibility
Integration with other tools	Depends on product and version.	Guaranteed integration with other products from their suite

Hosting an application can be a tricky business, choosing the right partner or solution can be a big business commitment and a long term one. There are multiple choices, including server virtualization, internal clouds, public clouds, external private clouds or Dedicated Servers.

We all know the traditional hosting solutions but what is a cloud provider?

A cloud provider is a company that offers component of cloud computing – normally Infrastructure as a Service (IAAS), Platform as a Service (PAAS) or Software as a Service (SAAS) – to other businesses or individuals.

Each solution has its own advantages and disadvantages. Maybe it is better to say each solution has its own focus. A dedicated server solution has an autonomy and control that can never be replicated by a hosting or cloud solution, simply put if the server is company property, you can put there whatever you want in any way you see fit you get to control: applications, runtimes, databases, storage, CPU capacity, and integration. Obviously there is a tradeoff, these solutions have scaling issues, and a considerable initial investment most importantly they force focus on areas that the business should probably not care about. The image bellow shows illustrates the compromise between solutions.

Figure 16 - Cloud Computing Paradigm Shift (Chou, 2009)

Cloud and hosted solutions brought a paradigm change in business control versus scale economy. So the question is what does business require and what do you want to focus on? The table below compares a normal cloud solution with dedicated server. To better understand the tradeoff between control and flexibility.

Table 5 - Cloud versus Dedicated Servers

	Cloud Servers	Dedicated Servers
Operating System	Support dozens of operating systems	New hardware may not work well with old operating systems and device drivers
Server Performance	Completely Scalable	High end expensive dedicated servers perform better than most cloud servers.
Hardware resource scaling	Can add CPU power and disk space while the server is running	Have to shut down the dedicated server and causing downtime when upgrading CPU or disk space
Server Snapshots	You can take a server snapshot first then make changes to the server	No snapshot feature
Hardware upgrade	When upgrade to a newer more powerful server, no need to reinstall OS or devices drivers. The same cloud server can run on different hardware	When upgrade to a newer more powerful server, need to reinstall OS and device drivers
Disaster Recovery	If your cloud server fails and is unrecoverable, you can simply power on the cloud server backup image. Your cloud server will be back and running in a few minutes	If your dedicated server fails and is unrecoverable, you will need to reinstall OS, restore files and databases from backups which can take many hours
Security	Provided by Supplier	Customizable
Privacy	Provided by Supplier	Customizable
Customization	Limited to None	Any Desired
Approximated Costs	$29/month to $199/month	$99/month to $1500/month

3 Agile Modeling

3.1 What is Agile Modeling?

Scott W. Ambler Definition (Ambler, 2001):

> *"Agile Modeling (AM) is a practice-based methodology for effective Modeling and documentation of software-based systems. Simply put, Agile Modeling (AM) is a collection of values, principles, and practices for Modeling software that can be applied on a software development project in an effective and light-weight manner."*

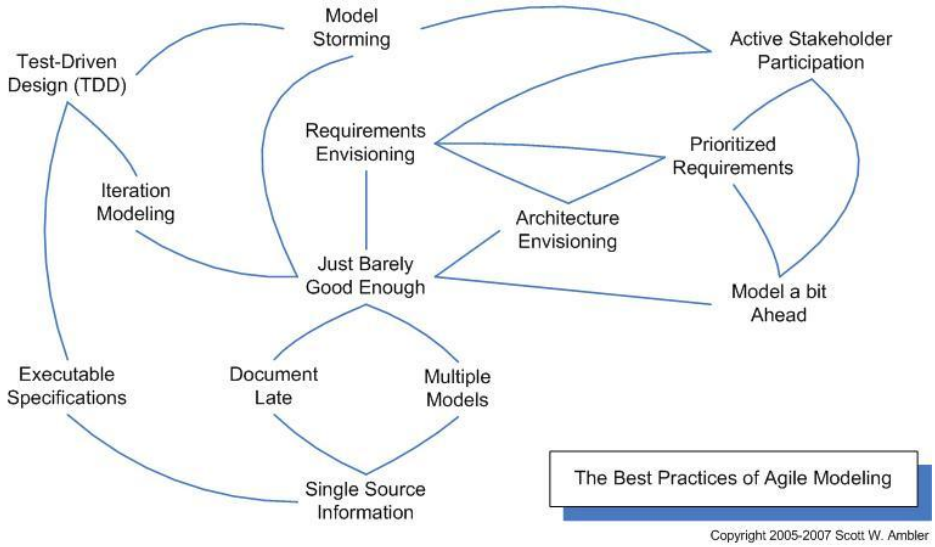

Figure 17 - Agile Modeling Best Practices (Ambler, 2001)

With a lite analysis, of the image above it is possible to get a pretty good idea of what Agile bases itself on, sentences like “Just Barely “, “Test-Driven Design” and “Model a bit”, clearly point to a development base methodology.

Agile by definition is the capacity of rapidly and efficiently adapt to changes. But what does Agile mean in a business concept? Agility means efficiency for the immediate future on the basis of action led pragmatism pushed to the extreme.

So the question remains, how does Agility, the necessity of constant adaptation and immediate results combine with a task commonly attributed to careful planning?

As a development methodology Agile, bases itself on a principle of simplicity and light-weight modeling so the main focus is development itself. Agile derives its concepts from Model Driven Development (MDD), with is one of the most commonly used development methodology nowadays. Here the specification of system functionality or functional requirement, are separated from the specification of its technological implementation. This is a commonly used process that allows for the separation of business consulting and requirement analyses, from the technological requirement and specification. Using this methodology a project begins trough a careful analyses of the business requirement which then results on a Functional Requirement Document (FRD), this document compiles the requirements and modules possible solutions using tools like Unified Modeling Language (UML), separating technical needs and constraints from business requirements completely. So before any coding is done, a number of analyses, meetings and modeling tasks are required, so that a final document can specify thoroughly what needs to be developed and how.

Agile bases itself on the same principles, with a simple but major difference. Instead of creating extensive models before writing source code, create agile models which are just barely good enough to drive overall development efforts.

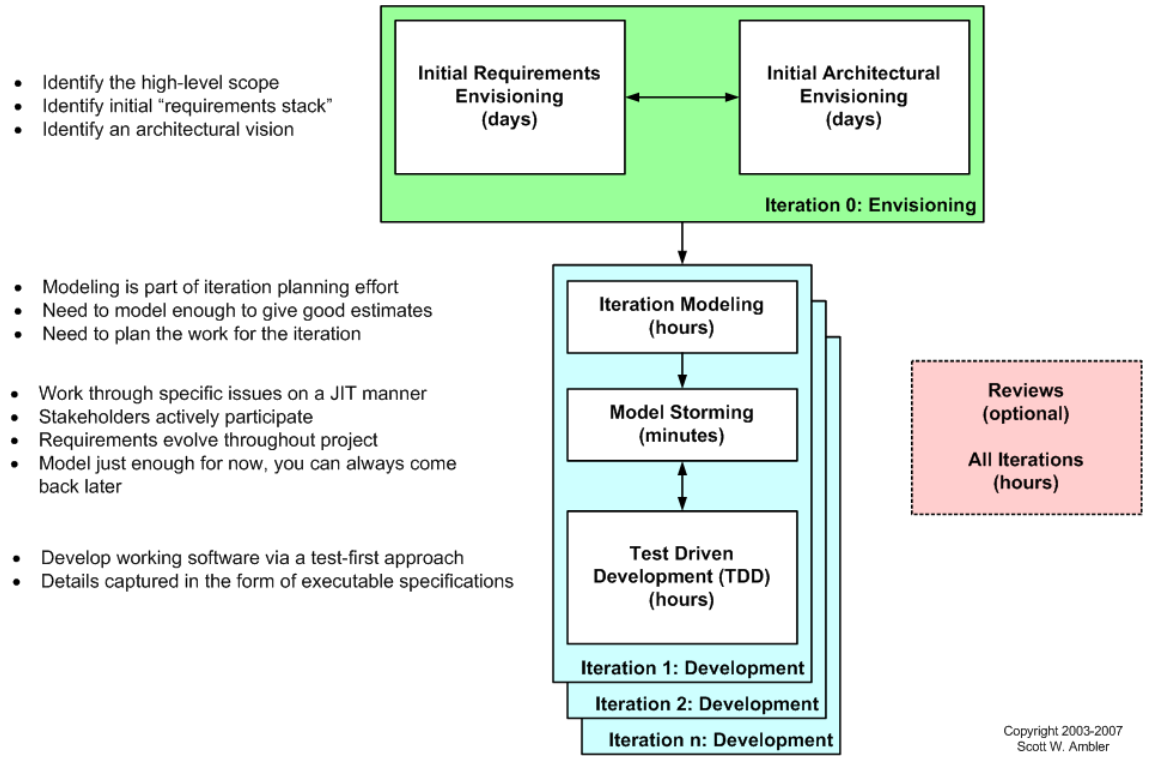

Figure 18 - Agile Model Driven Development (Ambler, 2001)

This difference vastly increases the focus on development, reducing the time and effort on initial documentation and analysis. Therefore the initial project emphasis turns to effort planning and time estimates, which need only to focus on high-level scope, general architectural vision and short term tasks and iterations.

3.2 Why Agile Modeling

People who develop software for a living quickly learns that, needs change and things don't always work as planned.

This is true for many and different reasons throughout a project lifecycle. Most common problems are well known and even well documented, problems like:

- Poor requirements definition
- Lack of Standards and Best Practices
- Miscommunication
- Unrealistic schedule

- Inadequate testing
- Adding features on middle of project development

Any project manager will tell you, that no matter how well planned and structured a project is problems will happen, a good one knows how to minimize them. Reducing problems has been a project manager struggle since the beginning and an adequate methodology goes a long way in solving most.

The advantages of methodologies are commonly known but what happens when the needs of a well design and structured plan, meet the reality of a small starting company. A company with multiple projects, that aim to build an entire user experience solution.

Here resource control and time productivity is vital, requirements are in constant change, and so is the product itself. In order to succeed a highly collaborative and evolutionary development approach is mandatory.

This is the advantage of Agile Methodologies, but what makes a development method Agile? (Pekka Abrahamsson, 2002):

- Incremental
- Cooperative
- Straightforward
- Adaptive

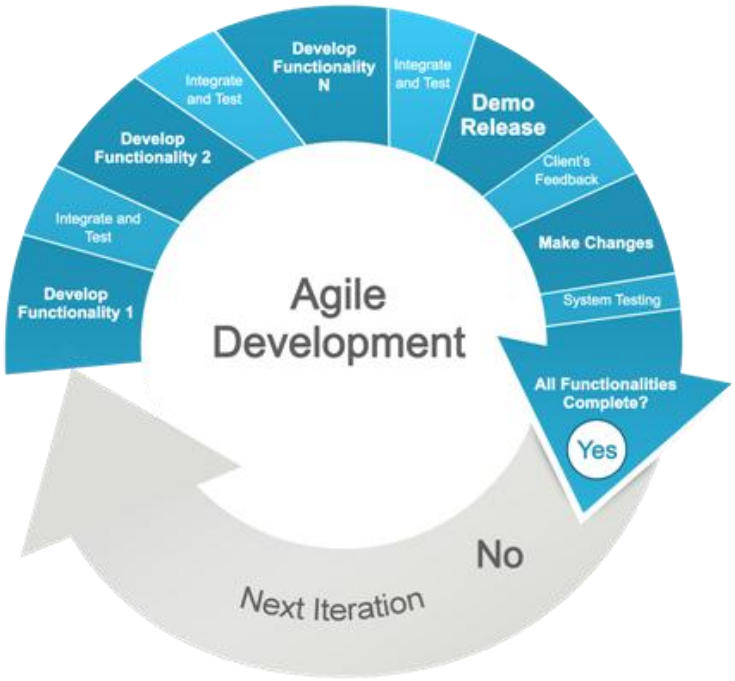

Figure 19 - Agile Lifecycle

Agile Methodology associates very well with the needs of a small company, as (Pekka Abrahamsson, 2002) defined. An agile methodology has to be Incremental, Cooperative, Straightforward and Adaptive. These attributes reflex perfectly what a small development team has to be in order to take a big project, especially when most of the project is still undefined.

These are some of the major benefits found during project:

- Agile helps speed up project beginning
- Reduces formal and lengthy meetings, informal meeting environment promotes team collaboration and participation
- Incorporates frequent and rapid changes into software development
- Allows for a task driven, leadership distribution
- Saves time and work by incrementally addressing work deliveries
- Allows early identification of problems and time deviations.
- Engages stakeholders continuously, better requirement gathering

- Increases team cohesion on delivery phases
- Flexible documentation policies
- Better tested and higher quality software

3.3 Agile Modeling in DW Construction (BEAM*)

In the data warehousing field, it is often hear if whether a person organization’s philosophy falls into Bill Inmon's camp or into Ralph Kimball's camp. These were the predecessors of Data Warehouse construction and are still responsible for the two major paradigms:

- Bill Inmon's paradigm: Data Warehouse is one part of the overall Business Intelligence system. An enterprise has one Data Warehouse, and Data Marts source their information from the Data Warehouse. In the Data Warehouse, information is stored in 3rd normal form.
- Ralph Kimball's paradigm: Data Warehouse is the conglomerate of all Data Marts within the enterprise. Information is always stored in the dimensional model.

There is no right or wrong between these two ideas, as they represent different data warehousing philosophies. In reality, the Data Warehouse in most enterprises is closer to Ralph Kimball's idea. This is because most Data Warehouses started out as a departmental effort, and hence they originated as a Data Mart. Only when more Data Marts are built later do they evolve into a Data Warehouse. (1keydata, 2012)

Agile Modeling although not stated is clearly better applied to Ralph Kimball´s philosophy. Bill Inmon´s method is characterized as a Top-Down approach. Here an overview off the all system is formulated, in order to gain insight into its compositional sub-systems by breaking down the overview to its base elements.

This is contrary to Agile Methodology that states:

- Do some initial architecture envisioning
- Model the details just in time (JIT)
- Prove the architecture early
- Focus on usage
- Don't get hung up on "the one truth"
- Organize work by requirements

- Active stakeholder participation

But how does Agile Modeling work and does it applies to Data Warehouse Development. A simplified vision of the process focus on four basic steps, supported by a Modeling method called BEAM*.

Figure 20 - Modeling a Data Warehouse with BEAM

Agile Dimensional Modeling or BEAM is a Modeling method for designing dimensional Data Warehouse and Data Marts. Keep in mind that agile focus on simplified and just enough documentation, so the Modeling methods are quite intuitive and more importantly quite common.

Table 6 - BEAM* Diagram Types

Diagram	Usage	Data Model Type
BEAM* Table	Modeling business events and dimensions	Business, Logical and Physical
Hierarchy Chart	Discovering hierarchical relationships, or prompting stakeholders for dimensional attributes	Business
Timeline	Exploring time relationships between business events. Used to discover when details, process sequences or fact duration	Business
Event Matrix	Documents the relationships between all the events and dimensions within a model	Business
Enhanced Star Schema	Individual dimensional models and generation physical database schemas	Logical, Physical

Although BEAM* diagrams are useful tools for documenting and structuring the same can be done with other modeling tools like UML or Relational Modeling Diagrams.

The true gain in Agile Modeling is in the way to conduct stakeholder meetings. Requirement phase is one off the most if not the most important stage in any project. A project with solid requirement facilitates the knowledge sharing between developing teams, allows for better delivery time estimates, more accurate task delegation and less time loss on restructuring or redoing work.

So how does agile DW contribute to solve a common requirement definition problem? Agile uses two main tools, Data Stories and the 7Ws Framework.

A data story is a lean requirement gathering technique. Data stories concentrate on data requirements and are written on whiteboards or spread sheets. Depending on how a story connects to time, it can be categorized on one of three types.

- **Discrete** - Short duration stories typically used to represent atomic-level events recorded by the operational system
    - A customer buys an item on a store
    - A customer enters a store
- **Evolving** - Longer running stories can take several days, weeks or months to complete. They represent irregular periods of time and the stories may or may not have finishes
    - A customer goes to a store and order a product and waits for it to be delivered
    - A Chain Directories orders a store audit and waits for the result
- **Recurring** - Occur at predictable intervals, useful to summarize discrete events but can also represent atomic detail for automatic measurements

These story types match the three types of fact tables available:

- **Transactional** - Inserted once never updated
- **Periodic Snapshots** - Corresponds to a predefined span of time, often a financial reporting period. The measured facts summarize activity during or at the end of the time span
- **Accumulating snapshots** - Corresponds to a predictable process that has a well-defined beginning and end. The grain of an accumulating snapshot for order processing, for example, is usually the line item on the order

Most business people do not fully comprehend modeling needs or how relational models work. On the other hand development teams commonly have narrow views of business events. Stories are facilitators. They help requirement definition by creating a common communication channel between business stakeholders and development teams.

This channel allow for modelers to build narratives that tease out the dimensional details of the business activity they need to measure. In order to do so, modelers ask questions using a simple 7Ws framework.

- **Who** - People and organizations
    - Example: Customers
- **What** - Things
    - Example: Products, services
- **Where** - Locations
    - Example: Store, warehouse
- **When** - Time
    - Example: Date/ Calendar
- **Why** - Reasons and causality
    - Example: Promotion, weather
- **How** - Transaction ID´s or status codes
    - Example: Order ID
- **How Many** - Measurements or Key Performance Indicators
    - Example: Sales revenue, stock turnover

The 7Ws Framework associated with stories help BEAM* modelers ask stakeholders the same type of questions that the stakeholders themselves will ask of the data warehouse when they become BI users.

Agile Dimensional Modeling doesn´t intend to completely replace traditional DW/BI analysis tasks, Agile Modeling makes traditional analysis tasks agile.

Agile Dimensional Modeling benefits:

- Focus on business processes rather than reports
- Creates flexible, report neutral designs

- Allows stakeholders to define business processes, from a measure perspective
- Iterative and incremental dimensional Modeling allows stakeholders to redefine their real data requirements
- Teaches stakeholders to think dimensionally
- Fully engages stakeholders in the design process, making them have pride in the final DW solution

In order to completely define a Data Mart, agile divides the action of into three distinct events (Figure 20).

Model Business Events

Business events are the measureable atomic details of business processes. (Corr & Stagnitto, 2011)

Understanding a business and its needs is a complex task. Agile DW helps modelers discover business events by asking a deceptive simple question.

Who does what?

The answer to the question is none other than an event. Basically the question is a short version of asking: Think of an interesting activity? Who does it? And what do they do?

This will engage stockholder into participating and more importantly to do it with confidence. The main reason is simple, were talking their language, business on subjects they know for years, how the companies work.

On the other hand modelers are receiving vital Modeling information, for example:

Question: Who does what?

Answer: Clients buy products.

This is the basic to start modeling a main clause. It is now possible to start filling a table trough continues iteration of the 7Ws Framework.

Table 7 - BEAM* Event Modeling Table Example

SUBJECT	OBJECT	DATE/TIME	LOCATION	QUANTITY	REASON	MANNER
[who]	**[what]**	**[when]**	**[where]**	**[how Many/Much]**	**[why]**	**[how]**
James	Pants	Today	Store	2 unit - 100 €	Promotions	P000234
Patricia	Apples	Last Month	Online	2kilos – 5$	New deal	P000123
…	…	…	…	…	…	…

The modeler must ask different stakeholder to participate different opinion make a richer event and has the story proceeds more information will be unfolded, also to better detail an event you can direct the meeting by applying themes.

- **Typical** - Your casual most common type of events
- **Different** - Useful to detect range of values, and otherwise hidden Modeling difficulties
- **Repeat** - Useful to detect what makes each story unique, granularity
- **Missing** - Used to detect were missing attributes may appear
- **Group** - Useful to detect variations in the meaning of details

Model Business Dimensions

Business events need dimensions to fully describe them for reporting purposes. (Corr & Stagnitto, 2011) Dimensions are data structures used to provide descriptions to atomic level events.

Having modeled business events there is no need for an elaborate discovery of dimension, has can be seen on the example (Table 7), dimension appear naturally:

- [Who] - Customers
- [What] - Products
- [When] - Date\Time
- [Where] -Location
- [Why] - Promotion Types

What remains is defining what attributes each dimension requires. This can be done in a much more direct way by simply asking direct questions:

- What attributes of [Dimension] would you like for reporting purposes?

- What attributes would you like to sort, group or filter [Dimension] on?

At this stage you can and should also resort to the 7Ws framework to help you better identify attributes, with questions like:

- **Who** is associated with a [Dimension]
- **What** is a [Dimension]
- **Where** are [Dimension]
- **Why** or **How** do they or you become a [Dimension]

It is basically asking: How would you like to organize your data? A dimensional Data Warehouse is only as good as its dimensions. Good dimension contain attributes that describe business using terms that are familiar and meaningful to the BI stakeholders.

Conformed dimensions allow measures from different events to be combined and compared. Simply put a conformed dimension is a single physical dimension table shared by several fact tables. Nonetheless separate dimensions can also be conformed dimensions if they are conformed at attribute level. There are three types of separate conformed dimensions:

- **Swappable Dimensions** - Subset dimensions. For example Supplier and Warranty Supplier. Warranty Supplier describes supplier the same way, but only for a percentage of the actual number suppliers
- **Rollup Dimensions** - Rollup has the name implies, are dimension with a same purpose but described at different levels, the most common example are provided by calendar dimensions and rollups like month
- **Role-Play Dimensions** - Single table dimension table, used for different roles. Calendar is also one of the most common examples, which can be used for sales date or supplier ordering date

Model Business Processes

A business processes is a sequence of multiple atomic business events combined. Most time stakeholders do not what to analyze a single event. They want to analyze a sequence of events. These sequences represent business processes and value chains.

Stakeholders model events chronological, after modeling a certain event you will notice that stakeholders will immediately start to associate it with the subsequent one. These sequences represent business processes that need to be measured end-to-end.

Sequence event can be Time/Value Sequence, basically small output actions came before high output events, for example before production there is procurement. Events can also be process driven, in other words a process driven event is a chronological sequence of events that have to occur in a specific order, for example a customer orders, warehouse picks product, a carrier delivers the product.

Table 8 - Event Matrix Example

Event	Customer	Employee	Carrier	Product	Product Type [rc]	Store	Delivery Address	Warehouse	Order_ID	Promotion	Shipment_Nr
	Who			What			Where		Why & how		
1. Customer Orders	*	√		√	√	√	*		*	√	
2. Product Shipments	√	√	√	√	√		√	√	√		*
3. Carrier Deliveries	√		√	√	√		√		√		√

*- Event creates dimension

√ - Event references dimension

An Event matrix is a vital tool, they help the modeler identify the most important events and conformed dimensions. They also allow the storyboarding of several events in just enough time. When comparing with standalone Data Mart projects, agile defends that more than one business model should be defined before development. There are several advantages in doing so:

- Identify patterns
- Identify potential valuable conformed dimensions
- Expose conformance issues
- Get stakeholders to conform inconsistent business terms up-front
- Build more robust ETL processes

Benefits

This allows the modeler to define business models while covering most of the situations business owners deal with every day, in a friendly non-technical way.

This method has several advantages:

- Meetings become more lively and participative
- Every point of view is heard
- People like to remember different situation
- Stockholders do not feel intimidated with technical terms like fact table or Entity relation connections
- Modeling several processes gives a wider vision of the customer needs

4 Solution

4.1 Milestones

Every solution in order to succeed must have defined goals and objectives. From project start a set of major milestones were defined in order to control and assess project evolution. The picture below describes the steps taken during the project lifecycle. Each of these is described in the next chapters.

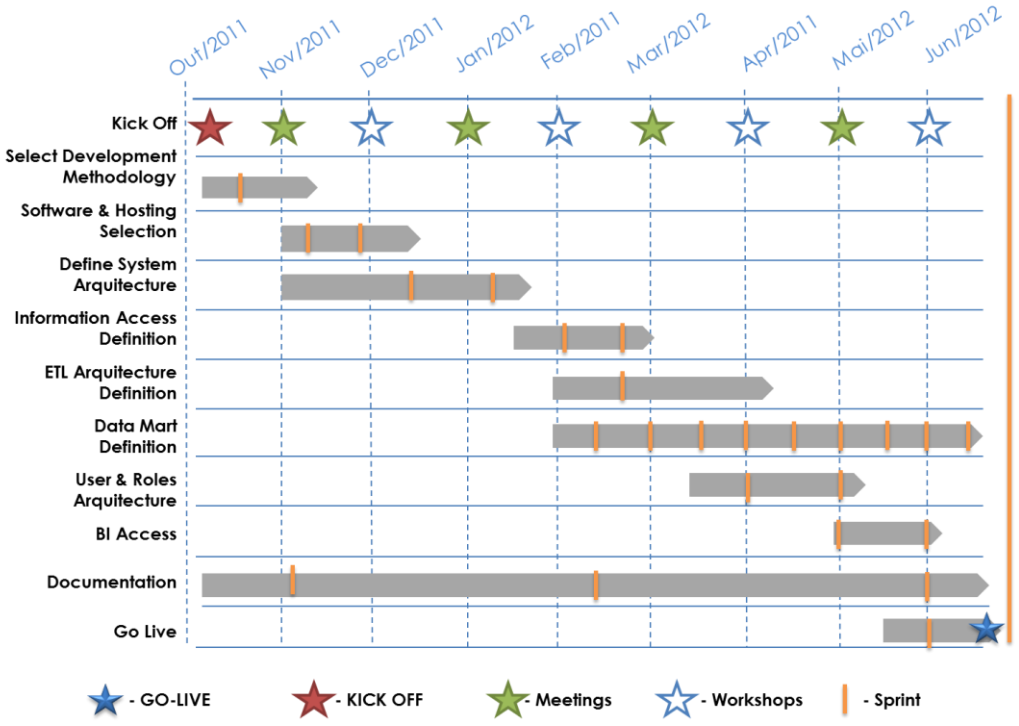

Figure 21 - Milestones, Sprints and Events

In conjunction with these milestones a set of Sprints were defined, a sprint is nothing more than a development iteration, and depending on the development complexity these sprints could be shorter or longer in time but approximately 15 days wide. Each sprint could have multiple objectives and more importantly at the end of each iteration a review process. This way development could be assessed in order to decide if it was concluded or needed further work.

In addition to the sprints a set of meetings and workshops were monthly scheduled. These internal company reunions had a mandatory obligation. Their main objective was to control project development from two different points.

Meetings intended to be status points, where future objectives were defined and previous ones evaluated. They also provided a chance for the involved teams to better comprehend and comment the project has a whole.

Workshops intended to be more technical gatherings. They aimed at several distinct objectives depending on necessity. Objectives like:

- Discuss model architecture
- Define component requirements
- Train users
- Define module parameterizations
- Performance issues

During the project lifespan other meetings internal and with customers occurred. These resulted of necessity and not of development methodology.

4.2 Solution Requisites

4.2.1 General

The focus is to build a Database System Architecture, capable of supporting the business requirements. This Architecture must be able to support an Operational Storage Area and a Data Warehouse which will feed Business Intelligence Requisites. The thesis focus is the construction of the Data Warehouse, therefore the defined objectives concern only the DW and vital areas directly connected.

A Data Warehouse is a database designed to support decision making in an organization. Data from operational databases is distributed to the data warehouse in order to provide a historical vision of operational information.

The following requisites as described were presented to the author at project start. They were described has the critical basic necessities. They were integrated in the document has to help describe what the Data Warehouse component had to achieve and from where it started.

The developed solutions for these requisites are modeled and described on chapter 4.5.5.

4.2.2 ETL Extract Transform and Load

The system must be able to import customer data, without concerning itself with the customer information system.

The system must control all information fluxes. This implies:

- **Error handling and notification** - Process must return error and warnings messages when execution fails
- **Non-integrated data registry** - Failed integration entries must be kept in a separated table for future analysis
- **Data validation** - All integrated information must be validated and homogenized
- **Clean up routines** - Information must be kept a maximum of three years, automatic process must ensure that unused or outdated information must be erased or kept in external files depending on each specific case
- **Step by step integration check points** - Processes must flag their execution allowing users to follow their progress from extraction to integration

The system must allow:

- **Different external sources** - Multiple client or information access methods like (FTP, Web services, files)
- **User defined integration tables** - User must be able parameterize the integration tables dynamically
- **Process dependencies** - Information dependencies must be kept on file execution, for example, it cannot be possible to execute a month consolidation if the daily consolidation for the referring month is not finished

- **Simultaneous integration** - Multiple area execution on nondependent areas, like sales and footfall
- **Unzip files prior to integration** - Extract zipped information to specific folder and posterior integration
- **Flow reprocessing** - Re-execute already finished processes due to late information entries or error situations

The system must allow flow modification like:

- **Create new flows** - Parameterize new integration flows
- **Edit an execution date** - Alter flow execution date
- **Adding or removing dependencies** - Alter process dependencies
- **Change data source** - Alter integration method or change integration file parameters like directory or file name
- **Change data final table** - Define information insertion table
- **Delete or inactivate** - Disable outdated or unnecessary flows

4.2.3 Data Warehouse - Data Marts

The Data Warehouse will be built using Star Schema Structure, and must support the following Data Marts:

- **Sales -** Customer Sales information, depending on client available information
- **Footfall -** People entrance counter. Number of clients that enter and exit a store
- **Objectives -** Client defined objective like Sales, Budget or Number of daily customers
- **Scheduling -** Employee Schedules, entrance and exit times
- **Atmospherics -** Store measured environment, CO2, temperature, energy consumptions and humidity
- **RESPS -** Movement tracking in stores, placement coordinates.

The table below describes the main attribute requisites presented to the author at project start.

Table 9 - Requirements - Data Marts

Area	Must comprise the following attributes	Dimensions
Sales	Quantity Net Value Gross Value	Date Employee Product Location Customer (If Exists)
Footfall	Number of Entrances	Date Location
Objectives	Value	Date Location Objective Type
Scheduling	Entrance Time Leaving Time	Date Location Employee
Atmospherics	Temperature Energy Consumption Humidity CO2 Levels	Date Location Sensor
RESPS	Mac address Class Name RSSI Antenna id Coordinates	Location Data Sensor

In order to support the previously mentioned areas, the following dimensions have to be created:

Table 10 - Requirements - Dimensions

Area	Must comprise the following attributes
Time	Time Hour 12 Hour 24 Minute Second Full hour 12 Full hour 24 Day period id Day period description

Date	Date
	Year
	Month id
	Month
	Month description
	Day
	Quarter id
	Quarter number
	Quarter description
	Full date
	Homologous id last year
	Week id
	Week number
	Week day number
	Week day description
	Fiscal year
	Fiscal month
	Sequence
	Is month last day
Employee	Employee
	Store
	Employee name
	Employee Company Number
	User
	Type
	Level
	Seniority
	Gender
	Date of birth
	Address
	Phone
	Email
	Parent employee
Locations	Location Id
	Hierarchical Detail Id
	Store
	Type
	Brand name
	Name
	Address
	Zip Code
	Phone
	Fax
	Email
Equipment	Equipment
	Name
	Description

	Type
	Parent Equipment
Product	Product
	Merchandise Detail Id
	Product Code
	Description
	Long Description
	Reference
	Color
	Dim
	Product group
	Season
	Supplier

4.3 Software and Hosting Server

The first step was the choice of software and hosting solutions. This is usually a sensitive subject and one with great impact on company roadmap.

Much of the analysis work done previous to product selection, can be found on chapter 2.3 - Software Tools and Hosting Services.

Below the author presents both choices made and the reasons for their selection.

4.3.1 Cloud Hosting – Microsoft Azure

Implementing the product on the cloud was the easiest project decision of all by far, the advantages were too great:

- Low initial investment
- Low maintenance
- Completely scalable
- No maintenance preoccupation
- Security protocols already implemented

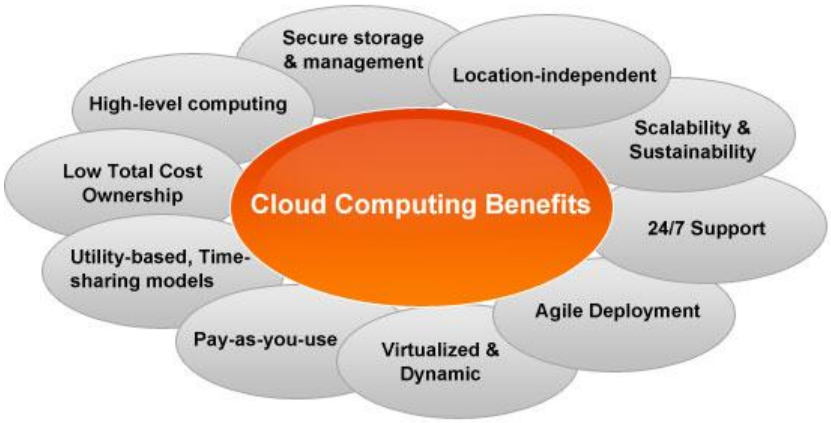

Figure 22 - Cloud Computing Benefits (Dot Com Infoway, 2012)

The only decision was which company to choose?

- Microsoft
- Amazon Web Services
- Joyent
- Salesforce
- Rackspace

Solutions were analyzed using the following criteria:

- **Reliability** - Existing client, support quality, already implemented solutions
- **Security** - How solutions and accounts were protected
- **Accessibility** - Deployment capability, database support
- **Price** - Hosting price, price by user access, bandwidth

Two points weighed heavily on the decision to Microsoft Azure Services, Microsoft partnership offer and Microsoft supported components, mainly SQL Azure.

The key components of the Azure Services platform include:

- Windows Azure
- SQL Azure

- Azure .NET services
- Live Services
- SharePoint & Dynamics CRM services

Highlights of Azure Cloud services:

- Secure, reliable application platform
- Safe data storage and hosting with 24/7 support
- Access to wide range of Microsoft's tools & technologies
- Dynamic, elastic and scalable solutions
- Utility model, pay-as-you-go facility
- Low Total Cost Ownership

Cloud simplifies the deployment process, whether implementing a website, batch routine or server application, they all work the same. Pick the solution, create a package and deploy to the cloud, no maintenance, no configuration.

“Compute provides three different role types that are designed to address needs specific to particular types of applications. All these roles run as VMs hosted in Azure data centers” (Tejada, 2012):

- **Web role** - Targeted to run web server applications, provides out-of-the-box support for Microsoft IIS 7 and ASP.NET. The Web role can also be used to host web services that rely on IIS
- **Worker role** - Targeted to broader, more common development of applications. Particularly used for executing background processing logic
- **VM role** - The VM role enables to upload a VM image of Microsoft Server 2008 containing the components you want to implement on Azure infrastructure. Regularly used for porting legacy applications

Why it was chosen:

- Low initial investment
- Perfect integration, with already chosen software development tools.

- Solution Deployment coverage with minimum parameterization
- Unified application control center
- Company's Microsoft Partnership
- Low maintenance work and costs
- Scalable
- Allows developed solutions to be easily exported to fix servers

4.3.2 BizSpark

Nowadays most development software tools, define the developed product market. Nevertheless the project responsibility was the selection of a SQL development Tool. SQL is more about selecting the language than the tool itself.

There is a wide range of SQL development languages, the main being:

- PostGreSQL
- MySQL
- SQL Server
- PL/SQL

The project required a tool that was company supported, capable of easy integration with multiple User Interfaces, be it mobile, web or others. A tool capable of supporting a SQL language that could:

- Supporting complex querying
- High information volumes
- Complex procedure implementation
- Capable of supporting an analytic/BI database

From the available SQL languages described above, only two satisfy all these requisites, SQL Server and PL/SQL. From a performance perspective and complex procedure implementation PL/SQL is the winner. This victory comes with a price, PL/SQL licenses are more expensive and from an accessibility point of view nothing beats SQL Server.

That being said there were three big factors that helped to make the decision:

- BI Development - Had to be taken in to account, and on this department Microsoft development tools associated with a Microsoft database are a perfect fit
- Chosen Hosting Solution - Even before choosing Microsoft Azure has the hosting solution, the choice for a Cloud based product was already taken
- Microsoft Partnership Offer - Microsoft offered a very interesting partnership, with access to all the necessary tools needed with very low investment

Figure 23 - Microsoft BizSpark Main Products

"Microsoft BizSpark is a global program that helps software start-ups succeed by giving them access to Microsoft software development tools, connecting them with key industry players, including investors, and providing marketing visibility to help entrepreneurs starting a business." (Microsoft, 2012)

Microsoft BizSpark made a compelling case, one that could not be matched by any competitor found.

Why it was chosen:

- Integrated development tools
- Several coding languages on same platform
- Development tools quality
- Commonly used by student
- Easy access to support information and tutorials
- Software quality
- Hosting solution chosen

4.4 Database Architecture

Database Architecture covers several areas not just how the servers are set up but also the components and the way they exchange information between them:

- **System** - Databases deployment and development architecture
- **SEEPlus data components** - Main application components and their roles
- **Information Access** - Internal and external communication between components

4.4.1 Systems Architecture

This is probably the most basic and commonly used database architecture for development and deployment. The Architecture consists in three different Database Servers or accounts since it is a Cloud solution. Each with its own specific roll easily identified by the account name:

- Development
- Tests
- Production

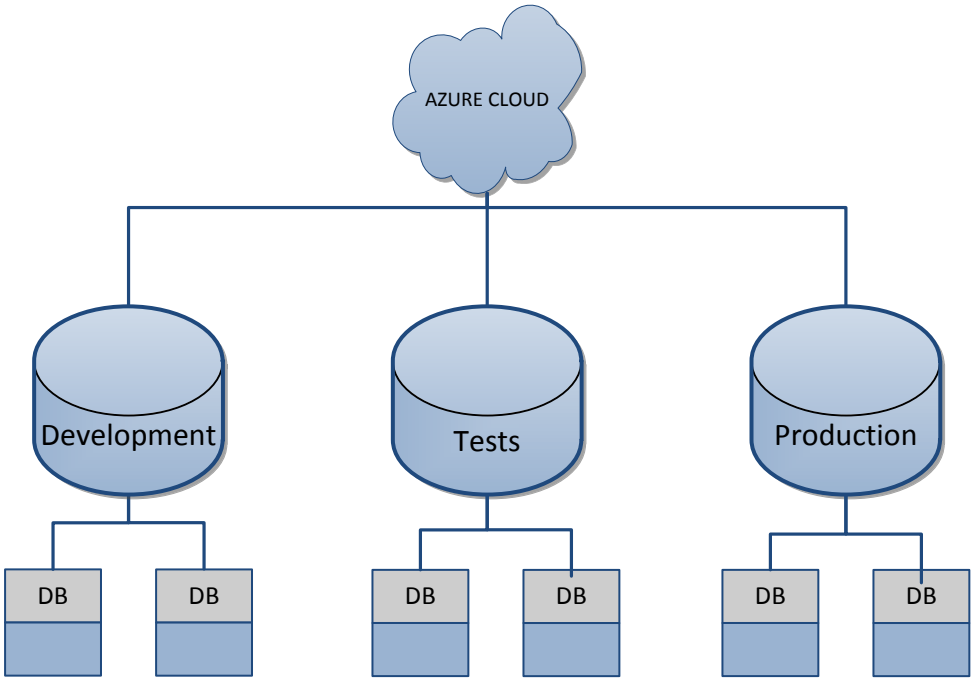

Figure 24 - Systems Architecture

As the name indicates development account, is used for development and experimentation. Here a user can develop freely without affecting anything. He can copy and create a database so that no impact is made on other people's developments.

Tests account has already client specific databases. Information differs from client to client, and sometimes so do the implemented modules. These particularities prevent the use of a common test database. Testing environment tries to replicate the existing conditions of a production server but in a smaller scale. This account is used for case testing, new developments or data validation problems. As a rule, the executed code like procedures or functions must already be in a certain degree of quality so it can be implemented in a Testing Database.

Production account is completely client driven. Each costumer has its own and reflects its current product version. All code must be properly developed and tested before entering a production database.

Database Deployment Methods

A cloud solution significantly simplifies production deployment, transforming it in a completely invisible process for the end user. Before deployment a system production copy can be made. All code alterations can then be made on the copy. This allows for a final validation in a real production environment, which after can be exchanged for the current, without impacting its users.

4.4.2 SEEPlus Data Components Architecture

The application architecture is based on the philosophy of full separation between extraction, transformation and loading routines and data area. Based on this principle, the block diagram presented bellow shows the three main components of the SEEPlus Application.

Each of these components has a different approach on how it deals with information.

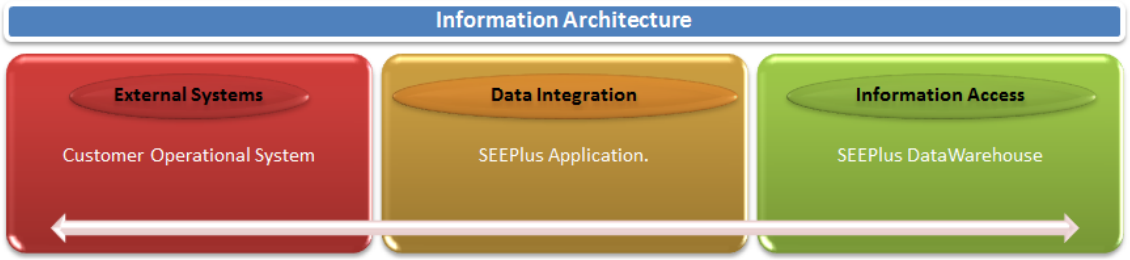

Figure 25 - Information Architecture

Customer Operational Systems is not a SEEPlus component nonetheless its relevance is major for the application as a whole.

Although SEEPlus has operational components, its core is not to be an operational system but more of an extension to the clients existing operational systems. Being so, external systems have a key importance as they are information sources. They provide data that will feed SEEPlus systems.

This information may regard any area like:

- Sales
- Stocks
- Human Resources
- People Counters (Footfall)
- Objectives
- Product Information
- Store Information
- Atmospherics (External Weather, Humidity, CO2, Temperature)

SEEPlus Application has the name implies is the major component, the heart. It's responsible for operational requirements and Data Staging Area (DSA).

Data Staging Area is an intermediate storage area between the sources of information and the data warehouse. It is responsible for:

- **External Access** - Gather data from different sources be it by Web Services, external Applications or FTP files integration
- **Information Gathering** - Quickly load information from various operational sources
- **Validation Processes** - Confirms user defined validation processes. Regularly business requirements
- **Data Cleansing** - Detecting and correcting (or removing) corrupt or inaccurate records from a record set
- **Attribute Unification -** Modify or restructure attributes in order to structure information under a common unified method

Operational requirement, are a set of necessary components and parameterizations in order for a product to work:

- **User Access Control** - Client validation credential. Roles and permissions according to customer purchased model or defined permissions
- **System Parameters** - Control UI needs, like System Data, value formatting Setting, language, system directories, libraries and many others

SEEPlus Data Warehouse is the final information layer. This component is responsible for all necessary exploration information needed by SEEPlus Interface, and like any Data Warehouse its focus is data storage.

Divided in several areas like Sales, Stocks, and Footfall is connected to various types of dimensions like Products and Stores, producing several Star Schema Data Marts.

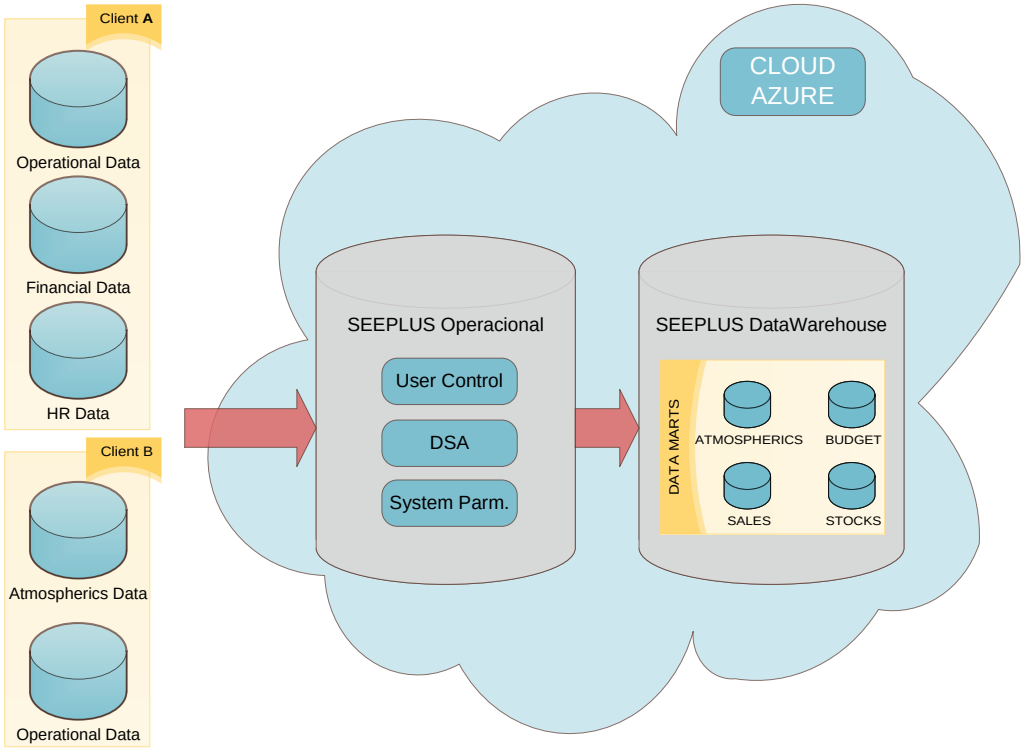

Figure 26 - SEEPlus Data Architecture

4.4.3 Information Access Architecture

Having multiple elements accessing information is a hard task to manage. More than that, having a complex system where different modules manipulate database information can become a disaster if not done properly.

Questions like who can do what? What module can access specific information? These interrogations become tricky to manage. Not all systems access information the same way or have the same requisites. Something as simple as user information is used throughout all SEEPlus suite. On mobile equipment, web pages or server applications, each with its own different validations and user required information.

Is it really possible to build different accesses for each user or application? Of course not. A never ending number of similar and redundant procedures would have to be built all with the same intent. Even worse would be to maintain all these procedures when an alteration became necessary.

So in order to guarantee a unified process, one that could be used throughout all systems independent of way of access or necessity, had to be built. This way no matter if an application wanted to delete, create or simply access a piece of information they would use the same defined method, guarantying uniqueness and data quality.

The defined architecture was built using a four layer approach:

- Lower Tier – Tables
- Middle Tier – Basic procedures to create, read, update and delete rows in tables
- Upper Tier – Communication layer between UI and table CRUD stored procedures
- Consolidation – Internal procedures used to create and join database information

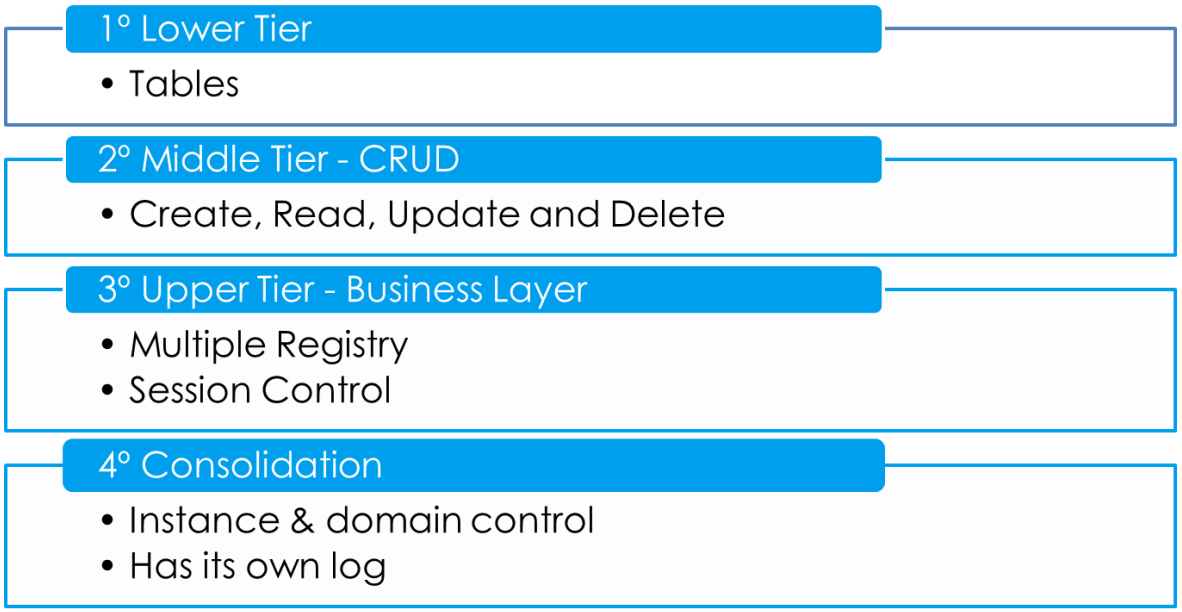

Figure 27 - Three Level Procedure Architecture

The first level or lower tier is the basic information storage, where information is keep, simply put the database tables.

The lower tier is the simplest and most basic necessity in keeping information in a table. Here a specific type of information is kept no matter its access needs or modification requirements.

A table is a set of data elements (values) that is organized using a model of vertical columns (which are identified by their name) and horizontal rows, the cell being the unit where a row and column intersect.

Middle Tier, is comprised of all possible actions associated with Data Manipulation Language (DML), for each combination of table and action, a stored procedure is created in order to manipulate it. All procedures receive the same two parameters. The first is a XML variable. Using XML has multiple advantages:

- Malleability
- Variable number of Attributes
- Minimum procedure change, when adding or removing attributes
- Fast processing
- Complex listing in simple manner

This allows the dynamic creation of SQL statements based on the XML information sent.

The second attribute is an Output variable with the single objective of returning a true or false flag indicating a correct or incorrect execution by the procedure.

The table below is an implemented Best Practice used to control procedural evolution. The table offers a quick view of the existing CRUD processes already implemented.

Table 11 - Middle Tier Control Table (Partial Table)

TABLE	SINGLE				Field Validation
	C	R	U	D	
[iCommon].[APP_CLIENT]	X		X		X
[iCommon].[APP_ENTITY]	X		X		X
[iCommon].[APP_ENTITY_CATALOG]	X		X		X
[iCommon].[APP_ENTITY_CATALOG_UDA]	X		X		X
[iCommon].[APP_HIERARCHY_VIEW]	X		X		X
[iCommon].[APP_HIERARCHY_VIEW_LEVEL]	X				X
[iCommon].[APP_MENU]	X		X		X

Upper Tier is used for outer communication with the database. Has previously stated the database can be accessed by many different systems. The Upper layer controls how that interaction is done. Usually this is called a business layer, since the main objective of this interaction is the execution of User Interface required task.

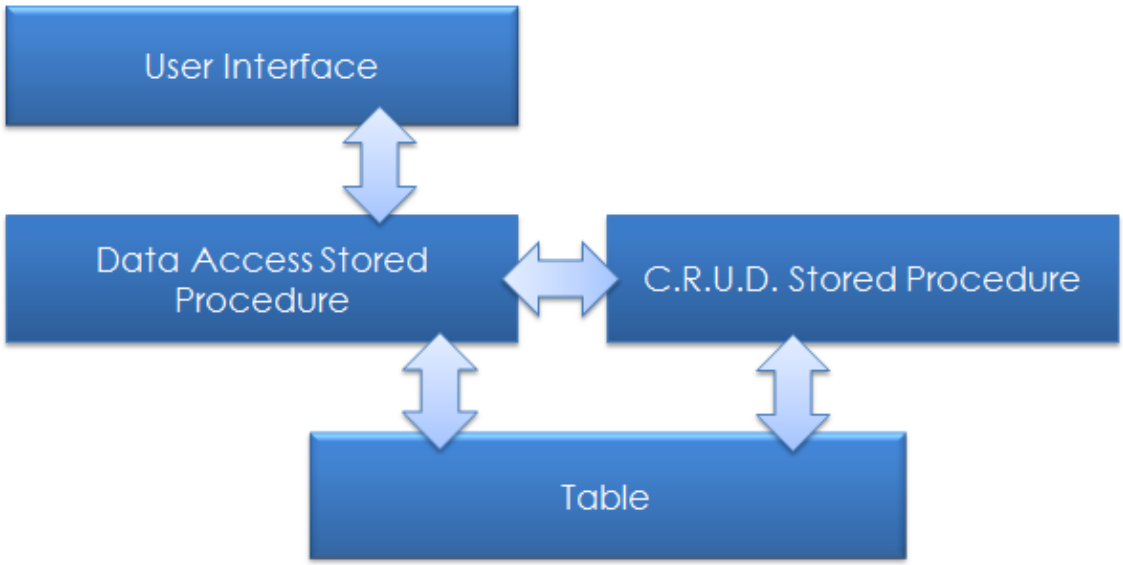

Figure 28 - User Interface Database Information access

These procedures contrary to middle tier are usually comprised by multiple actions, instead of a single one. They contain specific linked instructions in order to satisfy business requirements. Commonly used in the execution of complex querying, returning information from several joined tables.

Consolidation is an internal database process associated with Data Warehouse component. Consolidation main objective is to process data in order to create new information sources

depending on the desired requisites. Consolidation is a major part of SEEPlus Business Intelligence process. It starts from simple Data Mart fact tables like sales and applies aggregation or combination functions in order to produce optimized data sources for user interface consumption.

4.5 Data Warehouse Modeling

Data Warehouse Modeling was the most lengthy and iterative part of the all project. Truth be told, when constructing a Data Warehouse, business event Modeling is the reason why Data Warehouses are built.

When the project started, a set of modeling requirement were made (chapter 4.2), it became quickly clear that these requirement were not enough, they represent a small fraction of what each of these areas had to be modeled into.

Much like agile encourages the first customer meeting were scheduled before any modeling or development was made. This meeting intended to be just that, modeling reunions with the objective of defining user requirements.

The first meeting was a little chaotic, the lack of familiarization with the methodology and the insufficient experience, made the meeting a little too complex for the stakeholder. This was an important milestone one that brought many lessons.

The first thing was that although agile DW development did help gather stakeholders input, it was somewhat overcomplicated for the project needs. The methodology itself is based on a simpler and less bureaucratic approach to DW development. Nevertheless the examples given are for major DW implementations for one specific customer. From an author point of view this makes perfect sense because it helps modelers solve problems like:

- Understand the needs of different company areas
- Filter excess of information
- Uniform requirements
- Conform models
- Select which areas need to be modeled first

These requisites did not match the project, and so the first meeting became too complicated not just for the modeler but also for the stakeholders involved. The solution was not intended for major retailers, more than that some of the requirement had already been defined by the product developers, and were part of the product roadmap.

Has previously stated Agile DW development methodology helps modelers develop custom DW solutions based on a specific client needs. The aim of this project is to build a DW product based not on a single customer but on the needs of many.

In order to develop the already defined Data Marts the following work structure was developed based on Agile DW Design.

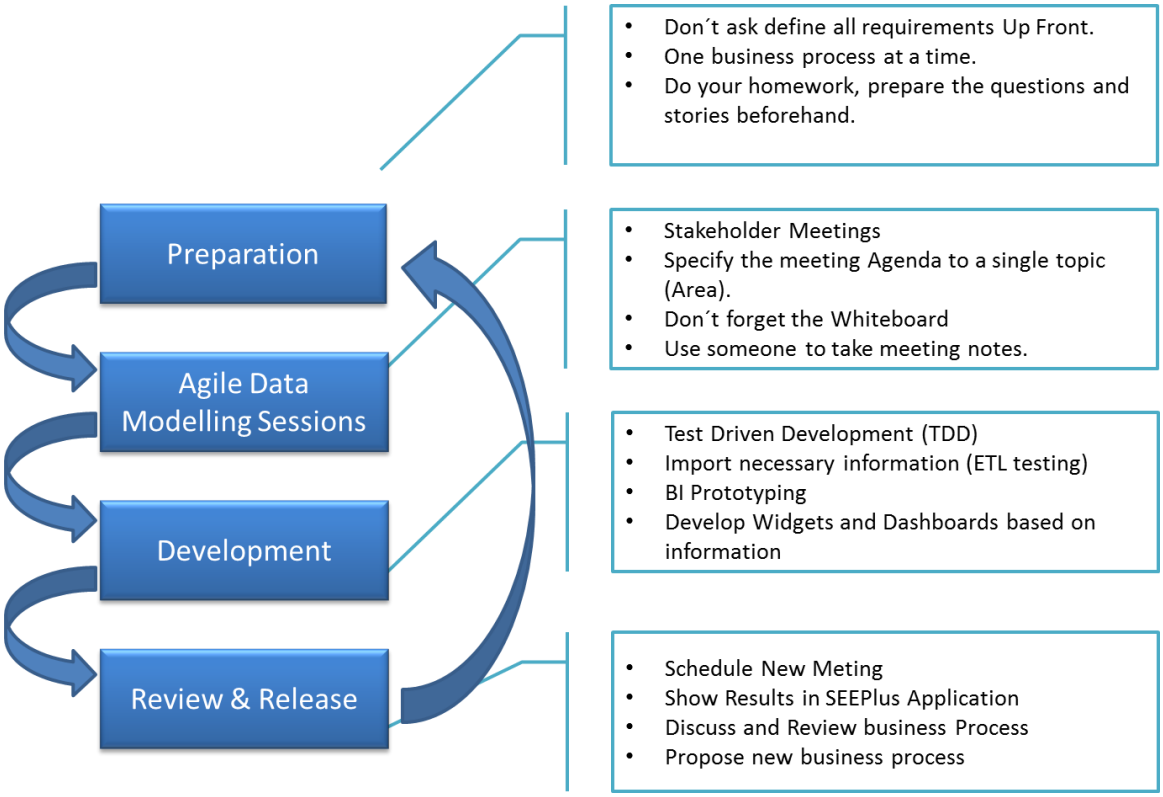

Figure 29 - InovRetail Data Mart Definition

4.5.1 Preparation

The first step before any modeling meeting is to define what needs to be achieved on that meeting. What are the goals and how do they align with the defined project requisites.

1. Scheduling stakeholder meeting
    a. Define agenda so that everyone is clear about the subject
    b. Make sure the stakeholders responsible for the decisions are present
    c. Prepare the necessary support needed equipment (Flipchart, projector)

2. Prepare the meeting

    a. Prepare some questions beforehand

    b. Prepare story examples to help start the meeting

    c. Ask a colleague if he can be present at the meeting to take notes

These steps may be simple and trivial but they can be the difference between a successful meeting were the objectives are accomplished and a meeting were every part involved leaves with more question than they went.

4.5.2 Agile Data Modeling Sessions

With a defined agenda and prepared meeting, modeling session ensued. These modeling sessions followed Agile DW Modeling methodology much like defined.

There were two slight differences, made to the methodology as seen on the picture below.

Figure 30 - InovRetail Agile DW Modeling

These are, define an Agenda before any event and gather Business Intelligence (BI) requisites at meeting conclusion.

Has mentioned before each session the agenda was defined, for example the first subject was sales. It reduced subject dispersion and specific theme modeling. It also permitted to model with several customers in a single timeframe. The advantages were clear:

- Defined subject matches company requisites
- Compare customer requisites on specific theme
- Better story and question preparation
- Create conformed views of the same business areas

The other difference was BI needs, during sales sessions it came to attention that although the clients presented different requirements, the story's told were quite similar. The stories evolved on

different ways but the intervenients were always the same. When faced with the question Who Does What? Whether the answer was a Client makes a purchase or an employee makes a sell. After developing the story the resulting modeled facts and dimensions were very similar. Sure there were differences, some clients calculated liquid sales with promotions others without, some did not store sales transactions at item level others did. But has meetings passed it was clear that Sales attributes were common to them all, with slight differences.

Dimension / Subject Area	Calendar	Equipment	Customer	Salesperson	Employee	Products	Locations	Suppliers	Mov. Types					Dimension Cou
Atmospherics														
TEMPERATURE	X	X							X					3
ENERGY	X	X							X					3
HUMIDITY	X	X							X					3
CO2	X	X							X					3
Resps														
RESPS	X	X							X					3
														0
Operational Information														
SALES A (i) (c)	X				X		X		X					4
SALES T (i) (c)	X			X	X			X	X			X		6
FOOTFALL (i) (c)	X								X					2
SCHEDULES (i)	X						X		X					3
STORE OBJECTIVES (i) (c)	X								X					2
														0
Dimension Usage Count	11			1		2		3		1		11		0	0	1	0	

Figure 31 - Logical Dimensional Matrix

So what made each customer unique? Why did they seem to have so different requisites? The answer became clear when the question made was, what do you use this information for?

The answer was Business metrics or Key Performance Indicators (KPI). These are value assessments that allow for performance measure of determined business activities or processes.

Depending on the business needs and the stakeholder business focus, sales information were used for many different purposes. This was particularly true between a retailer that has product production and a franchised store retailer. When modeling facts and dimensions, there were no differences between them. Both used the same Who´s and Why´s but when confronted with the BI requirements they could not be more different. While one focused on operation and logistical metrics the other gave a huge importance to human resources.

In order to better understand business differences it was added a new step at the end of each meeting, focused not on business processes but on business metrics. This allowed the gathering of a huge set of metrics based on each of the modeled areas. The table below represents some of the most important metrics gathered during these meetings.

Table 12 - Agile KPI's gathering

Area	Metrics
Sales	• Homologous Net Sales • Homologous Gross Sales • Homologous Quantity • Average Net Values by last 7, 14,21 and 28 days • Average Gross Values by last 7, 14,21 and 28 days • Average Quantity by last 7, 14,21 and 28 days • Sum Net Values by last 7, 14,21 and 28 days • Sum Gross Values by last 7, 14,21 and 28 days • Sum Quantity by last 7, 14,21 and 28 days • Max and Min Net Values by last 7, 14,21 and 28 days • Max and Min Gross Values by last 7, 14,21 and 28 days • Max and Min Quantity by last 7, 14,21 and 28 days • Net Value Week to Date • Gross Value Week to Date • Quantity Week to Date • Net Value Month to Date • Gross Value Month to Date • Quantity Month to Date • Net Value Year to Date • Gross Value Year to Date • Quantity Year to Date • Homologous Net Value Week to Date • Homologous Gross Value Week to Date • Homologous Quantity Week to Date • Homologous Net Value Month to Date • Homologous Gross Value Month to Date • Homologous Quantity Month to Date • Homologous Net Value Year to Date • Homologous Gross Value Year to Date • Homologous Quantity Year to Date • Week Value by W-1,W-2,W-3,W-4 • Aggregated Week Value by W-2,W-3,W-4 • Average Week Value by W-1,W-2,W-3,W-4 • Average Aggregated Week Value by W-2,W-3,W-4 • Homologous Week Value by W-1,W-2,W-3,W-4 • Homologous Aggregated Week Value by W-2,W-3,W-4 • Homologous Average Week Value by W-1,W-2,W-3,W-4 • Homologous Average Aggregated Week Value by W-2,W-3,W-4

Footfall	• Homologous Number of Entrances • Average Number of Entrances by 7, 14,21 and 28 days • Sum Number of Entrances by 7, 14,21 and 28 days • Max and Min Number of Entrances by 7, 14,21 and 28 days • Number of Entrances Week to Date • Number of Entrances Month to Date • Number of Entrances Year to Date • Homologous Number of Entrances Week to Date • Homologous Number of Entrances Month to Date • Homologous Number of Entrances Year to Date • Week Value by W-1,W-2,W-3,W-4 • Aggregated Week Value by W-2,W-3,W-4 • Average Week Value by W-1,W-2,W-3,W-4 • Average Aggregated Week Value by W-2,W-3,W-4 • Homologous Week Value by W-1,W-2,W-3,W-4 • Homologous Aggregated Week Value by W-2,W-3,W-4 • Homologous Average Week Value by W-1,W-2,W-3,W-4 • Homologous Average Aggregated Week Value by W-2,W-3,W-4
Sales Objectives	• Homologous Net Value Objective • Average Net Value Objective by 7, 14,21 and 28 days • Net Value Objective by 7, 14,21 and 28 days • Max and Min Net Value Objective by 7, 14,21 and 28 days • Net Value Objective Week to Date • Net Value Objective Month to Date • Net Value Objective Year to Date • Homologous Net Value Objective Week to Date • Homologous Net Value Objective Month to Date • Homologous Net Value Objective Year to Date • Week Value by W-1,W-2,W-3,W-4 • Aggregated Week Value by W-2,W-3,W-4 • Average Week Value by W-1,W-2,W-3,W-4 • Average Aggregated Week Value by W-2,W-3,W-4 • Homologous Week Value by W-1,W-2,W-3,W-4 • Homologous Aggregated Week Value by W-2,W-3,W-4 • Homologous Average Week Value by W-1,W-2,W-3,W-4 • Homologous Average Aggregated Week Value by W-2,W-3,W-4
Scheduling	• Total Seconds • Total Minutes • Total Hour • Total Time • Client Value

Atmospherics	• Homologous Temperature • Average Temperature by 7, 14,21 and 28 days • Max and Min Temperature by 7, 14,21 and 28 days • Homologous Humidity • Average Humidity by 7, 14,21 and 28 days • Max and Min Humidity by 7, 14,21 and 28 days • Homologous CO2 • Average CO2 by 7, 14,21 and 28 days • Max and Min CO2 by 7, 14,21 and 28 days • Homologous Energy Consumption • Average Energy Consumption by 7, 14,21 and 28 days • Max and Min Energy Consumption by 7, 14,21 and 28 days

4.5.3 Development

If all the requisites are identified and models of the necessary areas exist, development should be straightforward. This is not quite true, although the advantages of having good planning and development models is unquestionable, development itself can be very troublesome.

Has mentioned Agile DW Development defends a Test Driven Development (TDD), in order to properly test a development, three components must be ready:

- ETL Process
- DW Dimensional and Fact table Development
- BI prototyping

ETL Process

The ETL process is a must in any Data Warehouse System, the ability to import data to DW tables is unquestionably of paramount importance. This allows customer information to be uploaded to the DW and therefore compared and evaluated.

- Compare if modeling meetings were reflected on the retrieved information
- Discover hidden problems unnoticed on modeling sessions
- Analyze if all the defined requisites are in fact possible to achieve with the clients existing information
- Evaluate data quality and define quality controls

- Information volume analysis
- Identify possible performance issues

A TDD defines that in order for a model to work it has to be tested. Simply put, a DW cannot be tested without information. The first step in any data warehouse regarding information is to develop an ETL structure capable of managing information integration. Its main goal is to allow the integration of any new business flow made available by a customer, independent of its origin, source or format. Has such an Integration and Consolidation Engine was built, the created engine is defined on Chapter 4.5.4.1 Integration and Consolidation Engine.

DW Dimensional and Fact table Development

Dimension Development was a very tricky issue, not because it was hard to identify the needed dimensions or their attributes. With Agile DW Dimensional modeling it quickly became clear the intervenients each business process used.

The problem was to manage multiple different hierarchies in the same system. Each customer depending on its organization defines a hierarchy structure based on their business vision. This is impossible to conform, these structure represent business organization itself and has such it is wrong to impose one for every customer. The solution is described on Chapter 4.5.4.2 Hierarchical Structures.

Compared to dimensional development fact tables were much simpler to develop. Technically speaking this was the less complicated and iterative off all components.

After gathering the requisites provided by the modeling session it was quite easy to define fact tables capable of storing sales, footfall, atmospherics, objectives and scheduling Tables independent of the customer. A single ORIGIN_ID attribute allowed multiple client information to be stored in the same sales or objectives table.

All Fact development were done using snowflake Data Mart structure, has presented on the image below.

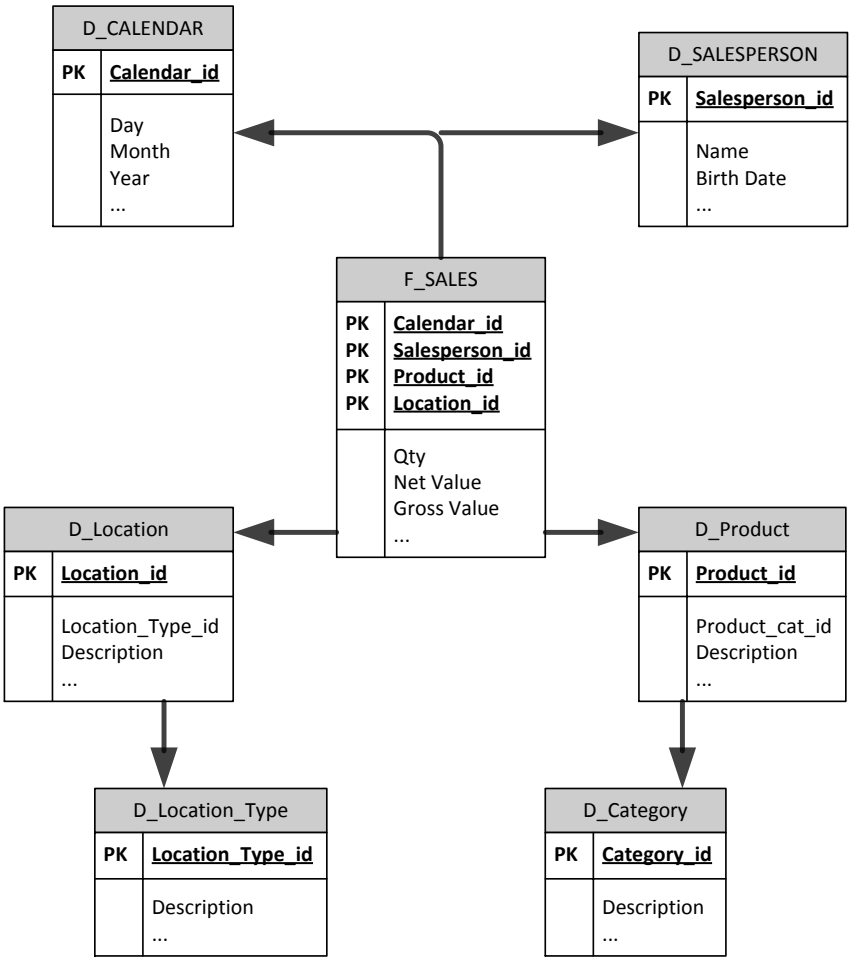

Figure 32 - Snowflake Module Example

BI Prototyping

The idea of BI prototyping is simple, has soon has you have something available to demonstrate, do it. After the development of the initial Data Marts and gathering of KPI´s, you now have all the needed information to start building BI models.

Much like Data Mart development, BI prototyping follows the similar stages:

- Define scope of prototype
- Design prototype screens / reports
- Load / convert prototype data
- Iteratively test prototype application functionality and develop

- Review / refine prototype with users

The scope was already chosen it derived from the Data Mart being developed. The information already existed in the system.

Initially the only existing way to present data was trough Microsoft Excel. Although it sufficed as proof of concept on earlier stages of development, it quickly became clear that it did not have enough quality to captivate or accurately describe what was intended to the customer.

But as iteration advanced, SEEPlus application started to develop several ways of access information:

- Table query interface
- Reports
- Interactive dashboards
- Widgets

Figure 33 - SEEPlus Widget Iterations FROM - TO

From:

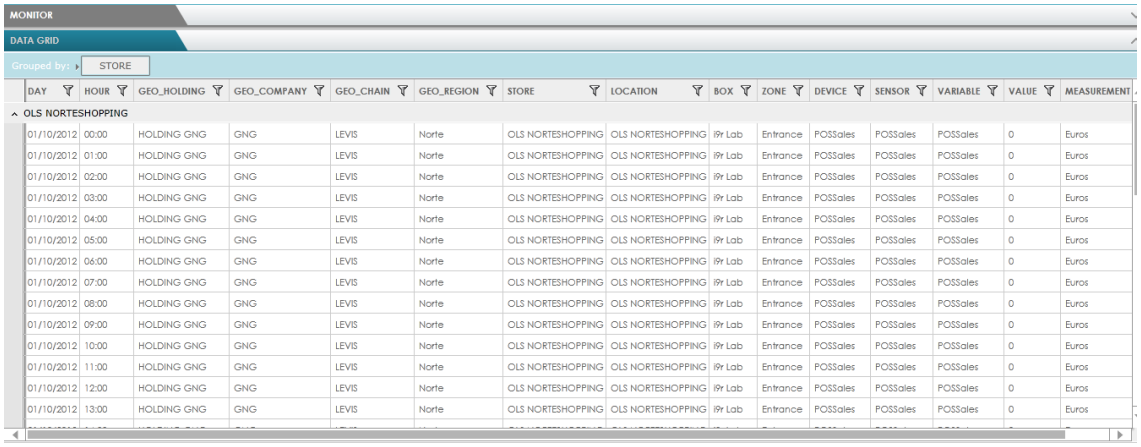

Figure 34 - SEEPlus Reports First Iteration

To:

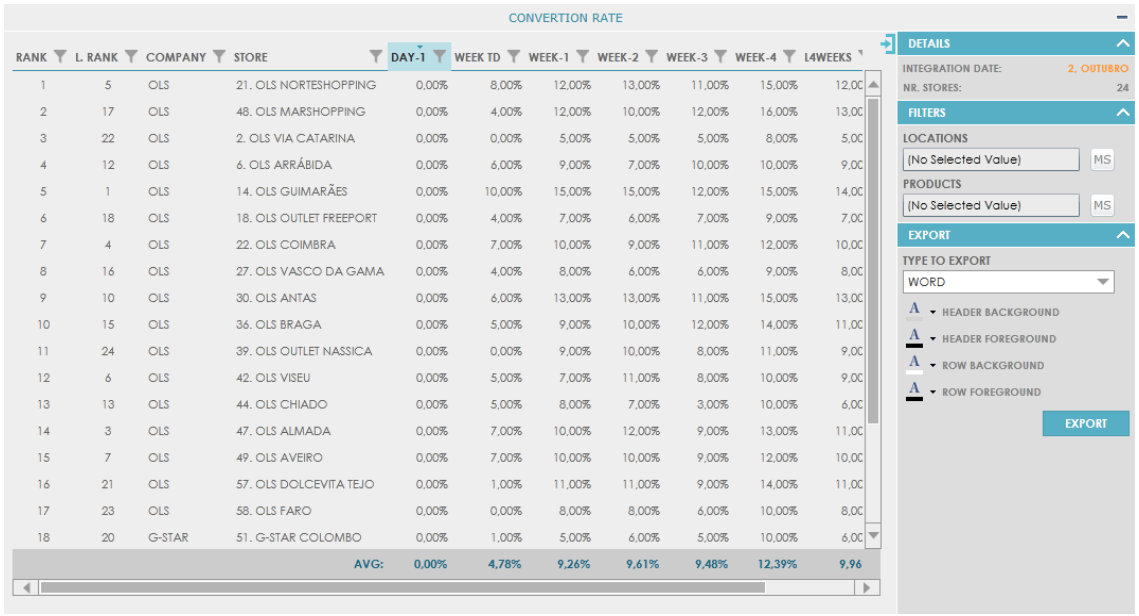

Figure 35 - SEEPlus Reports Current Iteration

The problem regarding BI prototyping especially in a completely new solution is that it takes time to develop. In an iterative environment where requirements keep changing this becomes near impossible to maintain.

From a UI point of view product development settles after a certain number of iteration. The component becomes static and shows information the Data Base provides. The same cannot be said for the procedures, tables and views created to feed those UI components.

In order to answer a set of requirements associated with dynamic display of different kinds of metrics, a framework named KPI Hub was developed. The solution aimed at solving the constants need for new information or modification of previously given one. The Developed solution is described on Chapter 4.5.4.3.

4.5.4 Key Modules

4.5.4.1 Integration and Consolidation Engine

"The Extract-Transform-Load (ETL) system is the foundation of the Data Warehouse. A properly designed ETL system extracts data from the source systems, enforces data quality and consistency standards, conforms data so that separate sources can be used together, and finally delivers data in a presentation-ready format so that application developers can build applications and end users can make decisions." (Kimball & Caserta, The DataWarehouse ETL ToolKit, 2004)

The importance of a good ETL system is paramount. In order to satisfy the needs and requirements presented the following Integration and Consolidation Engine was built.

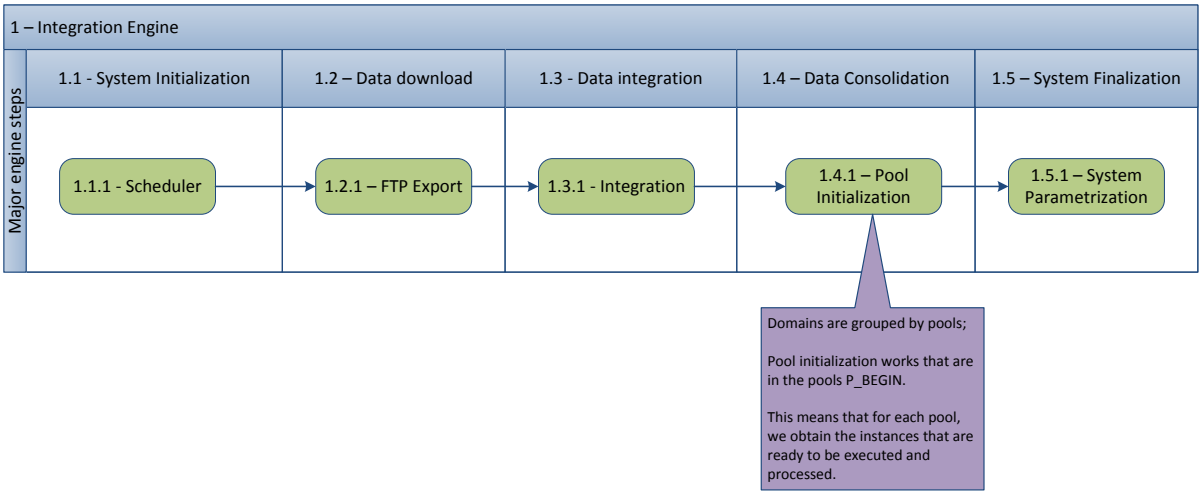

Figure 36 - I&C Engine Structure

As presented on the image above the Engine is divided in five distinct parts, each responsible for a specific stage of system information. Each process is controlled by a PowerShell script responsible for execution, log and control of the necessary procedures associated with the specific stage. A main PowerShell script controls the modular execution of each portion and is responsible for the execution of the next, once the previous is finished.

- System Initialization
- Data Download

- Data Integration
- Data Consolidation
- System Parameterization

System Initializations

Has the name implies this process is responsible for the information start up schedule. Based on Unix Cron, system initialization is a SQL time-based job scheduler.

System Initialization is based on a single SQL table, were all necessary domains are parameterized.

PROCESS_ID	PROCESS_DESC	PROCESS_CODE	PROCESS_TYPE	PROCESS_STATUS	MONDAY	WEDNESDAY	SUNDAY	SCH_DAY	SCH_MONTH	SCH_HOUR	SCH_MINUTE
1	GNG SALES AGG	2	INTEGRATION	SHEDULED	1	1	1	NULL	NULL	2	0
2	GNG SALES TKT	3	INTEGRATION	SHEDULED	1	1	1	NULL	NULL	2	0
3	GNG FOOTFALL	4	INTEGRATION	SHEDULED	1	1	1	NULL	NULL	2	0
4	GNG SCHEDULE	5	INTEGRATION	SHEDULED	1	1	1	NULL	NULL	2	0
5	GNG AGE GROUP	6	INTEGRATION	SHEDULED	1	1	1	NULL	NULL	2	0

Figure 37 - Engine Scheduler Table

The image above describes some parameterized domains, when analyzing a specific example like Process ID 1 or Aggregated Sales, the table shows that this process executes every day, note that all weekdays attributes are active = 1. Also SCH_DAY and SCH_MOTNH are empty, this fields allow the parameterization of specific execution on given day, month or both with translates to a single yearly execution. It's also possible to define the time of day the process will start by defining SCH_HOUR and SCH_MINUTE. As can be seen all processes are running daily and starting at 02:00 AM.

Data Download

Data download consist in downloading information from external sources into Azure Server. At this stage there is no information treatment or validation. SEEPlus does not impose an integration method. Although this offers great flexibility and the possibility of integration with any product, it also forces the application to work with Integration from multiple different sources.

This procedure is responsible for:

- **Retrieve information in specific folder or FTP** - Access external source and import to Azure
- **Extracting information if file is zipped** - Extract file from Zip file
- **Insert information into IDS schema** - Insert into respective Staging table
- **Manage File library** - Move files from Received, Error, Integrated folders
- **Update engine status** - Alter Flow status from Begin, to Working, unzipping, Error or Finished

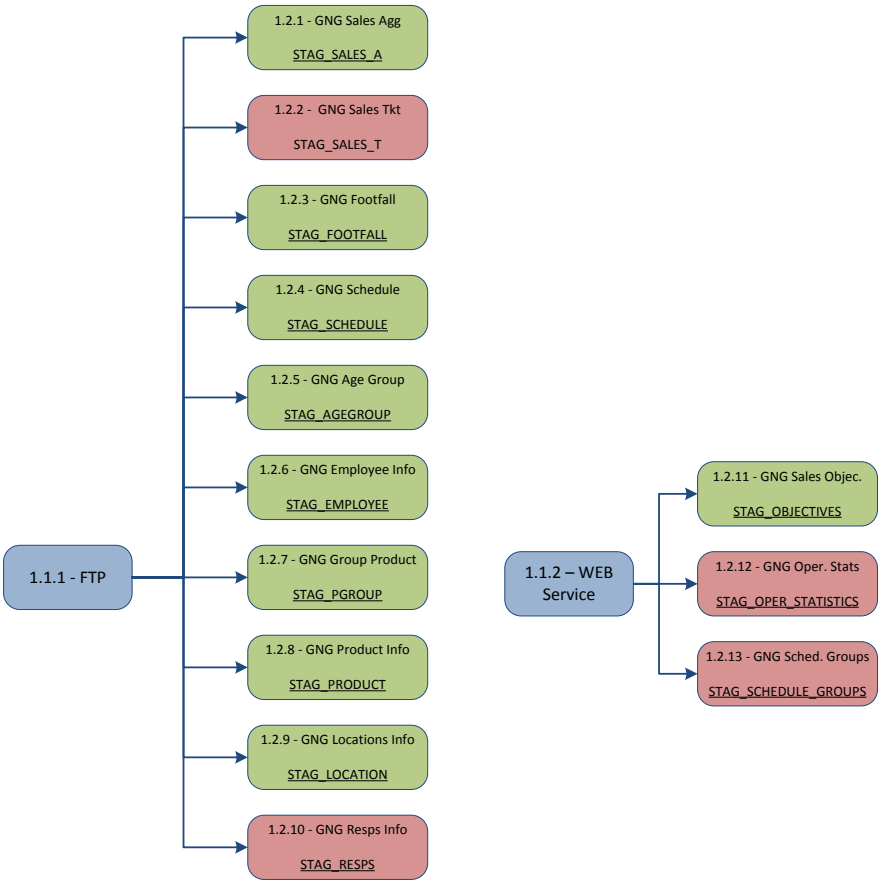

Figure 38 - Engine Data Download

In order to organize and unify all integrations, independent of its origin an IDS Schema was developed and is responsible for receiving all external information into system tables.

This process is controlled by a single table “ENGINE_DOMAIN” where domains are attributed to each integration flow.

Data Integration

Data integration is responsible for factual and dimension loading. This process is not only responsible for inserting information into DW tables but also:

- **Validate Information** - Nulls in mandatory fields, duplicate records
- **Data Cleaning** - Missing fields, Incorrect attribute types
- **Uniform raw information data into meaningful attributes** - Restructure information for example: MAN, M or MASCULINE into 501001 key field that refers to Masculine Gender

- **Guarantee Dimensional Information** - Experience tell that sometimes stores sell what they don´t have, product are sometimes sold and only a few days later registered in the system. We decided not to stop this sale from entering the DW but instead create a temporary product so that the sale can enter the system and maintain dimensional constraint

- **Update Engine Status** - Alter Flow status from Received, to Integrated, Working or Error

The image below describes the integration processes currently in the system. On the left the Staging Tables to the right integration tables, joined by the process responsible for the actions described above.

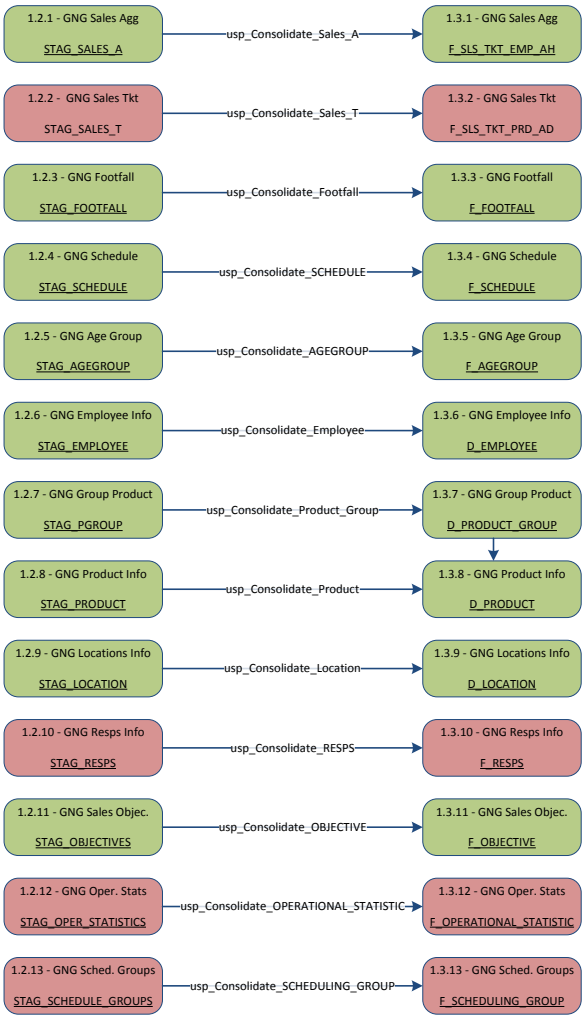

Figure 39 - Engine Integration Process

Data consolidation controls system aggregation. A consolidation results from an aggregation or union of a single or multiple fact tables. For example, transform a Sales ticket table into a Monthly Sales table.

Data consolidation is in most general cases a performance necessity, the most common possible example is the following:

Imagine there is a Fact Sales Table, it represent the thinnest sales grain, so as a standard it will have Location, Day, Time, Ticket, Product, Employee, Customer, Net Price, margin, Tax and several other. Now imagine a store that makes approximately 100 sales a day. This translates into something like 36.500 records a year. Now imagine the customer has more than one store that makes 36.500 multiplied by the number of stores.

A BI user does usually need all the sales detail. If a user wanted to see how much he sold in a year, each of those records would have to be aggregated real time in order to return the yearly value. Now if a Sales Monthly consolidation Table exists, it is possible to aggregate the necessary information into 12 records. These provide the exact same result with much lower processing cost.

Consolidation tables are an everyday need, one with rules, dependencies and constant validation. In order to create an automatic process capable of creating and maintaining consolidation tables, a consolidation engine was created. The picture below illustrates part of the existing process dependencies currently in the system. Each line represents a dependency that connects a parent process (left) to a child (right).

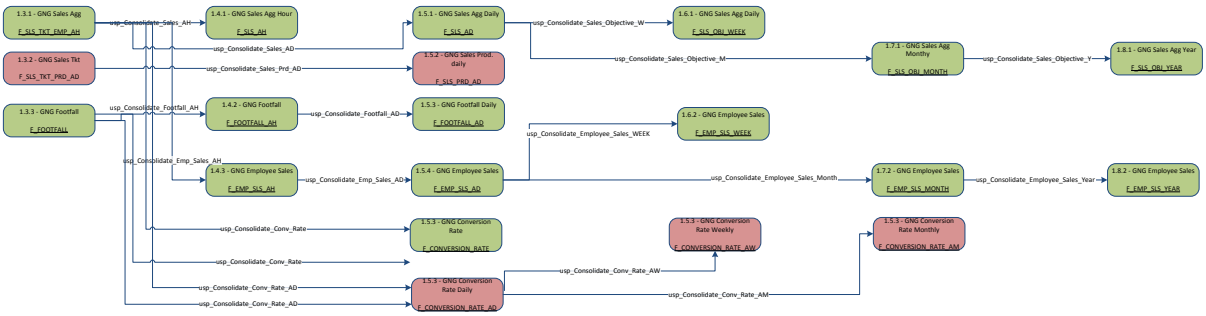

Figure 40 - I&C Consolidation dependencies

Much like integration flows, consolidation processes need to be parameterized on ENGINE_DOMAIN. Consolidation also needs to have a dependency, a procedure or procedure that will be responsible for triggering the consolidation. This way we can guarantee that a monthly process is executed only after the conclusion of its daily process. This is guaranteed by a ENGINE_DEPENDENCIES table.

System Parameterization

System parameterization has the name implies are a small group of parameterizations that are automatically altered after the integration and consolidation processes come to a conclusion.

- SYSTEM DAY - Validates date of last integrated information, and changes system date accordingly

4.5.4.2 Hierarchical Structures

A hierarchy is a flexible organized structure, arranged in a hierarchy form that usually consist of a singular or group levels/units. Hierarchies are usually client specific. Each client defines and structures information as appropriate, usually translating into their business vision.

Let´s take for example Organization A and Organization B, after a Dimensional modeling session it was defined that their merchandize hierarchies are represented has such.

Organization A - Company; Group; Category; Product

Organization B - Holding; Company; Group, Sub-group; Gender; Product

A normal DW could never join the information from both companies in the same repository. Organization A has a 4 level structure while Organization B has a 6 level. Even worse is the fact that the levels are not even the same, for example Organization B does not have Category. In order to store multiple client information and more importantly multiple hierarchies, a special dimensional structure had to be created.

The module below describes how any hierarchical structure is built in SEEPlus, whether it’s a Merchandize, Organizational or Equipment hierarchies.

The model is composed of five main components:

- **HIERARCHY_HEADER** - Existing Hierarchical Levels
- **HIERARCHY_DETAIL** - Items in each Hierarchical level
- **APP_ENTITY_CATALOG** - Hierarchy levels description and type
- **HIERARCHY_UDA_VALUE** - Relational table between Attributes and Details
- **D_UDA** - Item attributes not mandatory

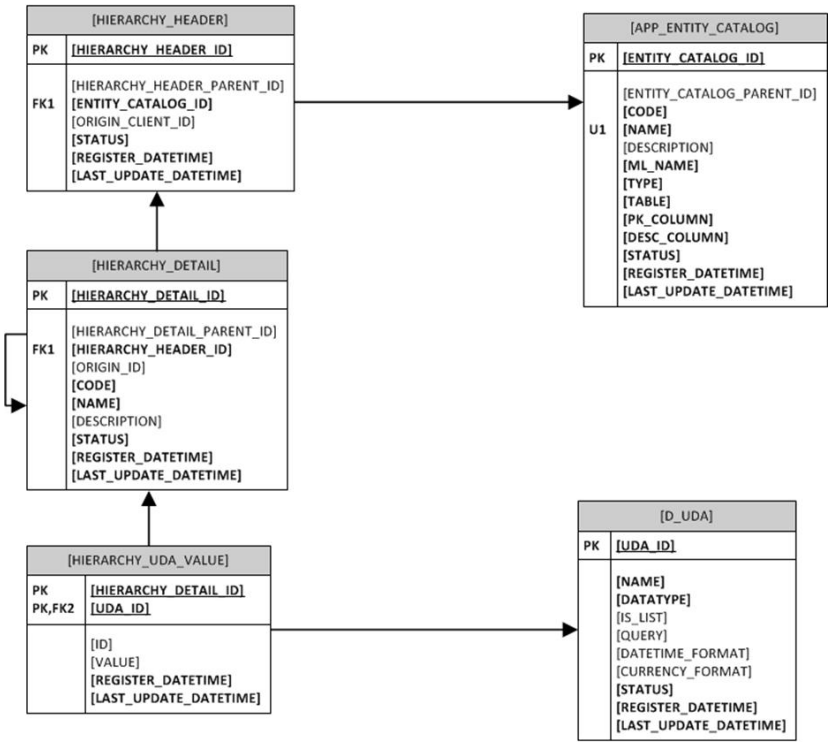

Figure 41 - Hierarchy Data Model

Looking at the example of company A, HEADER table would stay with 4 records each referring to a merchandize level. Now for each record inserted in header a single record would have to be added on the DETAIL table at least. So for example the first record associated with the level company could be “GNG” or “INOVRETAIL”, if for example a insertion was being made regarding a record associated with Category level, it could refer to something like “PANTS” or “IOGURTS”.

Just so that it becomes clear attributes are not added into the Detail Table, just the name and description of the category, item or company being represented. The reason for so is simple, Category and Company are completely different dimensions and has such have completely different attributes.

In order to solve this problem the UDA table or User Defined Attributes table was created. Here you can add any attribute and make it depend of any detail for a certain level.

In order to facilitate comprehension and navigation, views were created allowing a normal vision of the described dimensions.

Advantages

- Clear hierarchy dependency between structures
- Easily altered hierarchical structures

- Improved navigation
- Multiple hierarchical structures for client or business area

4.5.4.3 KPI Hub

This module was developed out of necessity due to the continuous imposed requisite alterations and the dynamic display of different kinds of metrics in the system.

The system is composed by four main tables and its branches:

- **APP_KPI** - Keeps necessary information in order to generate dynamic query
- **APP_MODULE** - SEEPlus existing Modules
- **APP_KPI_MODULE** - Defines what metrics can be accessed by each module
- **APP_VIEW_MODE** - Defines how a metric can be used (Charts, Lists, Grids)
- **REL_MODULE_VIEW_MODE** - Relational table between view modes and modules
- **REL_KPI_VIEW_MODE** - Relational table between view modes and metrics
- **APP_KPI_GROUP** - Metric associated group, (Sales, Atmospherics)
- **APP_KPI_EXCLUSION** - Which metrics cannot be used with another

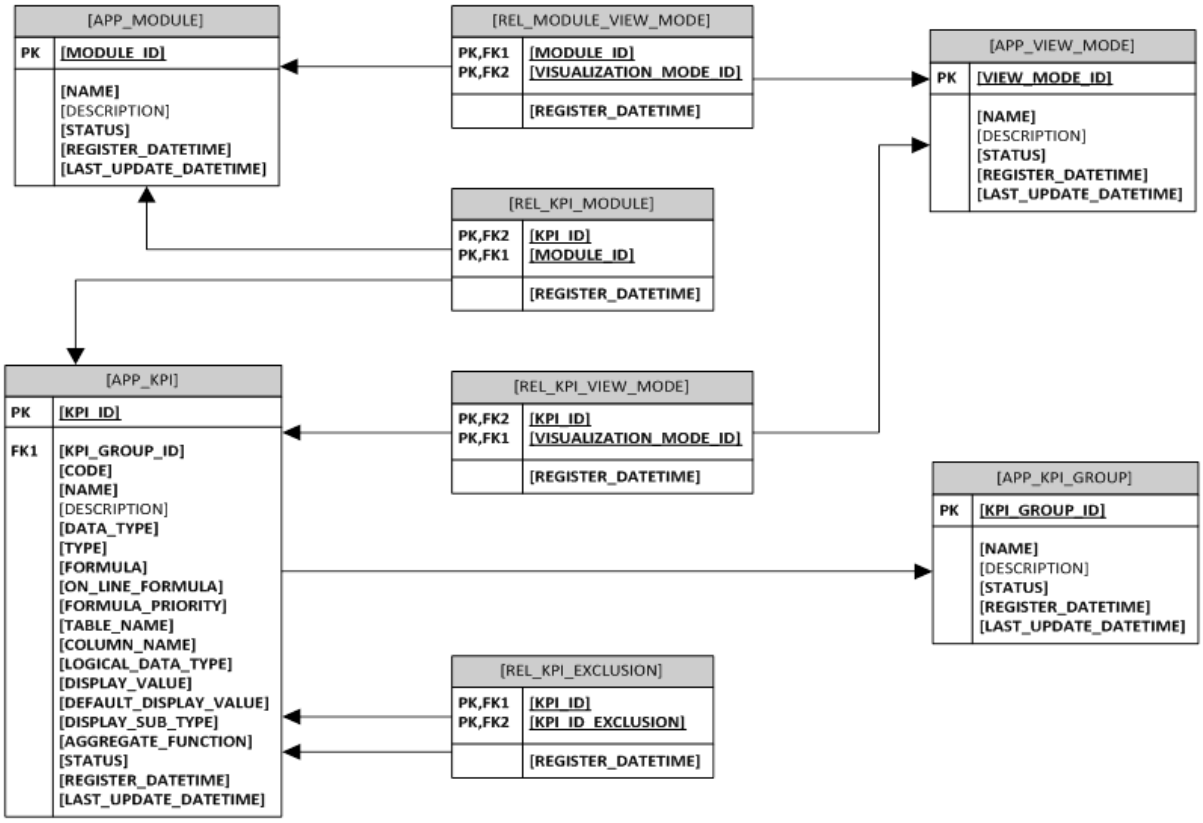

Figure 42 - KPI Data Model

This module aimed at solving two current BI problems:

1. Information needs keep changing and the system must be able to adapt without needing to alter tables, procedures or views

2. The same information must be coherent throughout the system

Basically the KPI HUB is a parameter based querying tool. A user defines its Information needs, how to calculate them, where to access the information origin tables or table. Based on that input a dynamic query is created. This query can now be invoked anywhere on the system and executed.

Advantages

- This model aims to create a dynamic infrastructure of creation and delivery of KPIs
- This model analyses any information existing in the system and trough SQL, builds blocks of information able to be used anywhere in the front end application
- Unified Metric Access Point
- Single point for metric alteration

- Module information control

4.5.4.4 Users

Like any secure software system, the implementation of user roles and access is a must. Simply put, through user registry it is possible to control application and information access. In order to apply these requisites to the SEEPlus application the following model was developed.

- **APP_SESSION** - Active system sessions
- **APP_USER** - User description table, containing user password and other relevant user information attributes
- **APP_USER_ROLE** - Relational table between users and roles
- **APP_ROLE** - User defined roles, several allowed (Administrator, Technician)
- **APP_ROLE_MENU** - Relational table between roles and menus defines visible menu options for a certain role
- **APP_MENU** - SEEPlus existing Menu options
- **APP_ROLE_MODULE** - Relational table between roles and modules
- **APP_MODULE** - SEEPlus existing Modules

The image below describes the implemented module used to store and manage users. More importantly, how information access is restricted to each user.

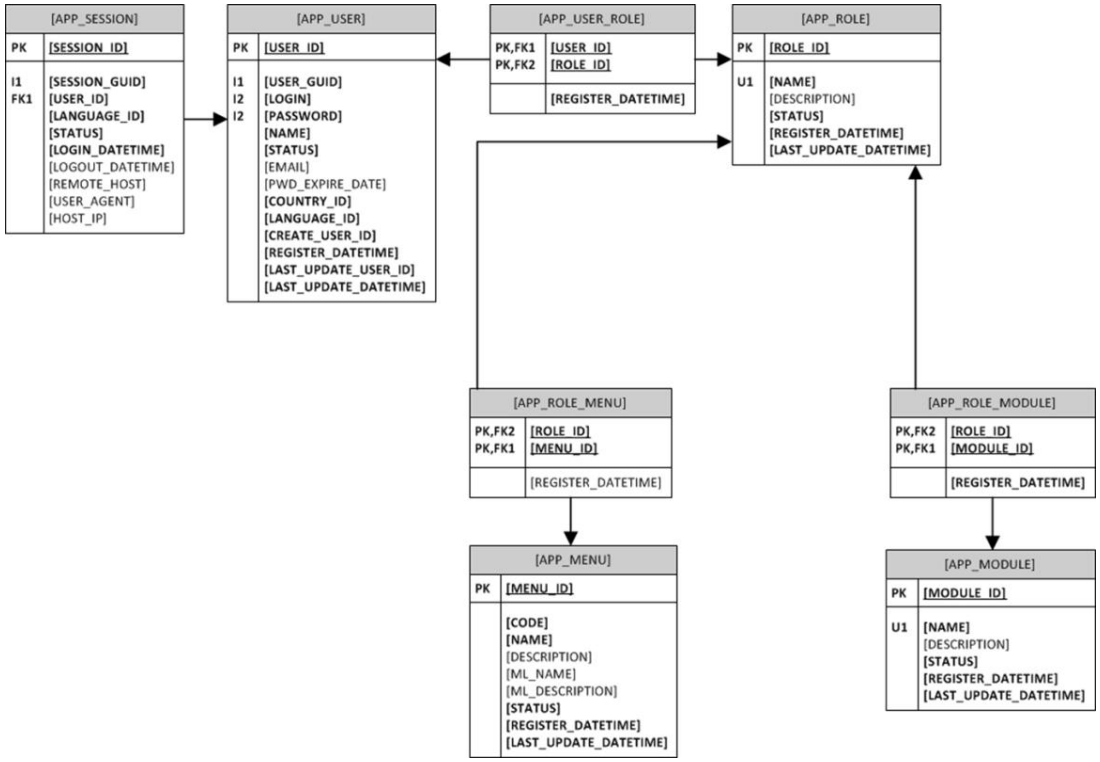

Figure 43 - User Data Model

Advantages

- **Control Application access** - Restrict or allow user to specific information
- **User Rastreability** - Determine who did what. Every business layer process stores user session information
- **Improved Data Security** - Control information access and protect client information
- **Minimize Data Inconsistency** - User must have access to perform data modification action
- **Improved Data Access** - Restrict number of users who can assess a given module at the same time
- **Role definitions permits** - Control multiple users through single common setup

4.5.5 Product

The developed data warehouse is currently implemented in more than one client. It is a continually evolving product, which has grown much further than was previously anticipated.

Has a result the data warehouse solution described in the document is currently feeding not one but three User Interface applications. Each of these application focuses on a specific business concept.

SEEPlus

Main application, a full set of modules aimed at improving store experience. Modules like:

- **System Cockpit –** Application control and parameterization: users & security administration, process integration control, define system variables, Foundation Data
- **Aim –** Real Time Business Reporting: business reporting, real time monitoring, advanced business dashboards, business alarmist, Map Navigation
- **Command Center –** Energy and Store Environment Control: atmospherics Strategy, Sensors Remote Control
- **Explorer –** Atmospherics Effects & Management: Atmospherics Alerts and Automatic Scenarios, Store Real Time KPI´s
- **Spaces –** Store Layout and movement analysis: Designer Layout Analytics, Advanced Visitor Forecasting
- **Optimizer –** Business Control: Measure initiatives, Advanced Customer Profiling, Optimize HR Allocation, GRID Analytics
- **Feeds –** Instant Customer Feedback: Create and deploy surveys on the fly, Analyze results

SEEPlus is an innovative and fully integrated solution, built to aid retailers in their endless effort to manage and optimize their retail environments.

SEEMobile

A mobile version of SEEPlus, interactive layout redesigned with mobile constraints in mind. This application is a lighter version of SEEPlus, contains some of the main modules but with different features associated. SEEMobile intends to be a more operational application, one capable of supporting initiatives triggered by SEEPlus.

- **Aim –** Real Time Business Reporting: business reporting, real time monitoring, advanced business dashboards
- **Feeds –** Instant Customer Feedback: Respond deployed surveys
- **Spaces –** Store Layout and movement analysis: Designer Layout Analytics

In Store Monitor (ISM)

A desktop application capable of delivering real time reports trough widget like screens.

In store monitor is an always on top desktop application, that allow stores to have a current business vision on how their doing. The image bellow illustrates some of the main presented features.

- **Atmospherics Info** – This module presents information regarding the three main environment variables, Temperature, CO2 and Humidity
- **Employee Performance** – Measures employee productivity, by several different factors like: Weekly Sales, Hours Worked, Client Value, Quantity Sold and others
- **Targets** – Store Objectives, percentage of completeness of objective like: Average basket size, basket value, total sales and conversion rate
- **Voice of Customer** – Feeds Inquiry results
- **Request Service** – Request store intervention, allows the selection of predefined type of problem. Allows the user to describe the issue and its details. This triggers a notification sent to the person responsible for said problem

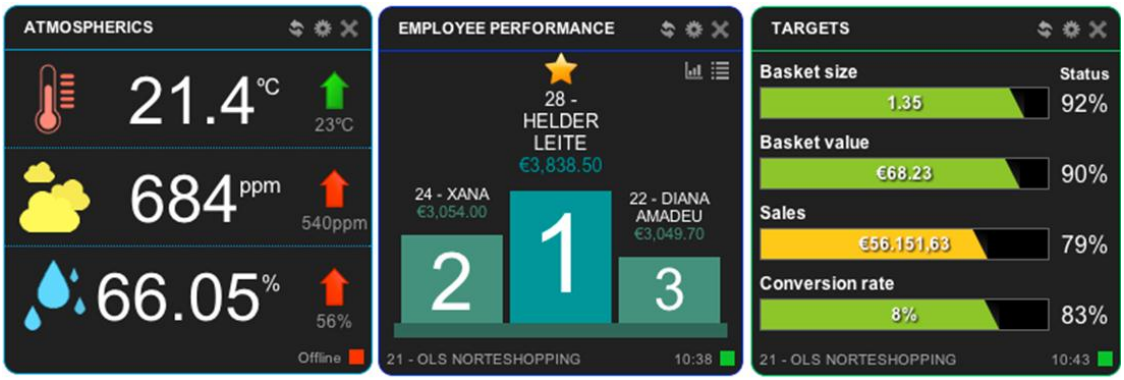

Figure 44 - ISM Widgets

4.5.6 Review and Release

Review and Release stage is accurately described by what the name implies. After development the results were analyzed. Evaluations are made by two different entities, the company and the customer.

For each development iteration, a Sprint was defined. This sprint, usually 15 days would start with the gathering of requirements and end with a meeting in order to evaluate the development made.

Client demos, intend to show work progress. Here the goal was not so much to define if the development was completed or not but to evaluate the clients response to the application. These demos are particularly useful to show the outcome of previous modeling and requisite gathering sessions.

Depending on the outcome of these meetings the goals defined by the new Sprint, could be the restructuring or improvement off the latest developments or the definition of the next ones.

4.5.7 Documentation

The present document started to be written by the end of June and continued until early October. During previous months many drafts, models and sketches were made. Some with more quality that other but always with the same intent, be good enough to get the concept understood, developed or appraised.

During the projects lifecycle there was a constant search and production of information. As common to any Agile methodology this information does not always translates into documentation.

The first months were the most prolific in analyses and state of the art research. This period required not development but decisions, be it to choose the development Software, the development methodology or product hosting solutions. This decisions implied search, evaluation, and comparison in order to decide.

Shortly after the choices made, a new preoccupation arose. The definition and structuring of the product, what was intended to do and what it strived for, compared to the competitors. Here product analysis and solution comparison helped find, where current DW/BI solutions are more fragile, their focus and intended customers.

5 Conclusion

Modeling and building a Data Warehouse is not an easy task. Building a robust and trustworthy one from nothing was a challenge like no other.

Agile Data Warehouse Modeling was a great source of information, one that considerably helped at project start up. Preparing stakeholders meetings, knowing what questions to ask or refining documentation and notations, it was a clear help on a massive set of tasks.

That being said Agile Modeling for DW focus on companies with a size and information sophistication level way above your common retailer. It also became clear that this phenomenon was not constricted to just methodologies, but also suppliers of DW/BI solutions. Data warehouse development and BI tools continue to focus on massive companies, companies with size and budget capable of supporting the projects described on Agile Modeling examples.

As a result Agile DW design is completely focused on client specific solutions, one project one client a specific solution. The project focus was quite different, to build a product centric DW that aims to serve multiple client needs.

Solution development was a lengthy and complex process. At start requisites were poorly defined and mostly generic. There was a lack of experience using Agile Methodologies in business requirement meetings. Constant changing requisites lead to a chaotic development. More importantly there was no apparent solution to some of the most crucial requirements, like allowing multiple customer hierarchies on same product.

During the project lifecycle, one by one all problems were solved. Experience made business meetings more productive and with it, requirements definition easier. Development best practices were defined which help structure and homogenize the project. Meetings, sprints and workshops helped tackle major problem by engaging them as a team. Client demos gave important customer feedback which was them discussed and transformed into possible new features.

These aspects helped built a robust and reliable Data Warehouse product. Has a result, the product is presently used by most of the customers that helped define and module what is still a constant growing product. During research no example or competitor was found capable of satisfying the project main needs. Create a Data Warehouse product for multiple retailers, capable of providing innovative analysis on customer retail experience and store environment. Making it has a truly unique product.

The developed solution involves continues work, with the addition of each new customer new requirements are defined and new Data Marts developed. Its already part of the product roadmap to extend to areas like stock and logistics. Nevertheless there is still much work to be done with the current developed Data Marts, with new models and new metrics being created on a daily basis, the necessity for new consolidation and aggregations are a constant. Product evolution is a constant and in a small start-up company what is priority today may not be tomorrow. Still the base is built, a robust and trustworthy Data Warehouse, capable of supporting any new business area the future might bring.

Bibliography

Dot Com Infoway. (2012). *Cloud Computing*. Retrieved August 31, 2012, from http://www.dotcominfoway.com: http://www.dotcominfoway.com/technology/cloud-computing

1keydata. (2012). *Bill Inmon vs. Ralph Kimball*. Retrieved September 23, 2012, from www.1keydata.com: http://www.1keydata.com/datawarehousing/inmon-kimball.html

Ambler, S. W. (2001). *http://www.agiledata.org/*. Retrieved January 28, 2012, from Agiledata: http://www.agiledata.org/

Andrews Consulting Group. (2011). Why You Need a Data Warehouse. *JD Edwards Advisor*, 8.

Beck, K. (2010). *Manifesto for Agile Software Development.* Agile Alliance.

Bednar, J., & Robertson, D. (2006). *Development Methodologies*. Retrieved 08 1, 2012, from www.inf.ed.ac.uk: http://www.inf.ed.ac.uk/teaching/courses/sapm/2005-2006/slides/methodologies_4up.pdf

Boehm, B. (2002). Get ready for agile methods. *IEEE Computer*, 64-69.

Chou, D. C. (2009, January 13). *Cloud Computing and the Microsoft Platform*. Retrieved August 31, 2012, from http://blogs.msdn.com: http://blogs.msdn.com/b/dachou/archive/2009/01/13/cloud-computing-and-the-microsoft-platform.aspx

Corr, L., & Stagnitto, J. (2011). *Agile Data Warehouse Design.* Leeds: Dec1sion Press.

EMA. (2010). *Top 10 Things You Should Know About IBM Cognos 10 for Business Analytics.* Enterprise Management Associates.

Feuerstein, S. (2010, January). *O'Reilly Community*. Retrieved Setember 06, 2012, from www.oreillynet.com: http://www.oreillynet.com/pub/au/344

Howard, Philip. (2011, August 31). *IBM Smart Analytics System vs. Oracle Exadata for Data Warehouse Environments.* Retrieved June 12, 2012, from smarterquestions: http://smarterquestions.org/2011/08/ibm-smart-analytics-system-vs-oracle-exadata-for-data-warehouse-environments/

IBM. (2012). *IBM*. Retrieved August 20, 2012, from http://www.ibm.com: http://www-01.ibm.com/software/data/industry-models/retail/

Inmon, B. (1992). *Building the Data Warehouse.* Wiley.

Intellitech Consulting Inc. (1997). Software Engineering Best Practices - Iterative Development. *Software Engineering Best Practices.* University of Albany.

itelligence. (2010). *Microsoft Business Intelligence*. Retrieved August 23, 2012, from www.contemporary.co.uk: http://www.contemporary.co.uk/products/microsoft-bi-products/

Kimball, R. (1996). *The Data Warehouse Toolkit.* Wiley.

Kimball, R., & Caserta, J. (2004). *The DataWarehouse ETL ToolKit.* Indianapolis: Wiley Publishing, Inc.

Kobielus, J. (2010, April 30). *What's Not BI? Oh, Don't Get Me Started....Oops Too Late...Here Goes....* Retrieved August 20, 2012, from http://blogs.forrester.com: http://blogs.forrester.com/james_kobielus/10-04-30-what%E2%80%99s_not_bi_oh_don%E2%80%99t_get_me_startedoops_too_latehere_goes

Kreisberg, S. (n.d.). *Retail Metrics: Key Performance Indicators (KPI's) – Stock to Sales Ratio*. Retrieved 08 10, 2012, from Onestepretail.com: http://onestepretail.com/retail/articles/2010/09/retail-metrics-key-performance-indicators-kpis-stock-to-sales-ratio.php

Larman, C. (2004). *Agile and Iterative Development: A Manager's Guide.* Addison-Wesley Professional.

Microsoft. (2012). *BizSpark*. Retrieved August 31, 2012, from http://www.microsoft.com/bizspark/: http://www.microsoft.com/bizspark/

Microsoft. (2012). *Cloud Computing*. Retrieved August 31, 2012, from http://www.microsoft.com: http://www.microsoft.com/enterprise/viewpoints/cloud-services/default.aspx#fbid=nzuSkoorHNl

Microsoft. (2012). *IT Cost Optimization*. Retrieved August 31, 2012, from http://www.microsoft.com: http://www.microsoft.com/enterprise/optimizing-costs/default.aspx#fbid=nzuSkoorHNl?Pillar=tilesWrap1

Microsoft. (2012). *SharePoint 2010*. Retrieved August 27, 2012, from www.microsoft.com: http://sharepoint.microsoft.com/en-us/Pages/default.aspx

Microsoft. (2012). *Why SQL Server*. Retrieved August 31, 2012, from http://www.microsoft.com: http://www.microsoft.com/sqlserver/en/us/product-info/why-sql-server.aspx

MicroStrategy Inc. (2012). *Business Intelligence for the Enterprise*. Retrieved August 31, 2012, from www.microstrategy.com: http://www.microstrategy.com/software/business-intelligence/

Oracle. (2008). *www.oracle.com*. Retrieved August 20, 2012, from www.oracle.com: http://www.oracle.com/us/industries/retail/046439.pdf

Oracle. (2009). *Oracle Reports Services Product Architecture.* Oracle.

Oracle. (2012). *Oracle Business Intelligence Enterprise Edition Plus*. Retrieved August 20, 2012, from www.Oracle.com: http://www.oracle.com/technetwork/middleware/bi-enterprise-edition/overview/index.html

Oracle. (2012). *Oracle Business Intelligence Suite Enterprise Edition Plus Components*. Retrieved August 15, 2012, from www.Oracle.com: http://www.oracle.com/technetwork/middleware/bi-foundation/enterprise-edition-platform-compone-093074.html

Oracle. (2012). *Oracle Reports Services 11g*. Retrieved August 20, 2012, from http://www.oracle.com: http://www.oracle.com/technetwork/middleware/reports/overview/index.html

Pekka Abrahamsson, O. S. (2002). Agile Software Development Methods. *VTT Publications*, 107.

Perks, M. (2003, September 29). *Best Practices for Software Development Projects*. Retrieved Setember 04, 2012, from http://www.computerworld.com: http://www.computerworld.com/s/article/85198/Best_Practices_for_Software_Development_Projects

Power, D. J. (2007, March 10). *A Brief History of Decision Support Systems.* Retrieved August 22, 2012, from http://dssresources.com: http://en.wikipedia.org/wiki/Business_intelligence#cite_note-power-1

Rud, O. P. (2009). *Business Intelligence Success Factors: Tools for Aligning Your Business in the Global Economy.* John Wiley & Sons.

SAP. (2012). *SAP Business Information Warehouse*. Retrieved May 16, 2012, from SAP: http://help.sap.com/saphelp_nw04/helpdata/en/b2/e50138fede083de10000009b38f8cf/content.htm

SAP. (2012). *SAP Business Information Warehouse*. Retrieved 2012, from help.sap.com: http://help.sap.com/saphelp_dimp50/helpdata/en/b2/e50138fede083de10000009b38f8cf/content.htm

SAP. (n.d.). *SAP Business Information Warehouse*. Retrieved 2012, from help.sap.com: http://help.sap.com/saphelp_dimp50/helpdata/en/b2/e50138fede083de10000009b38f8cf/content.htm

Tejada, Z. (2012, March 21). *Windows Azure Development: An Architectural Overview*. Retrieved August 31, 2012, from http://www.devproconnections.com: http://www.devproconnections.com/article/windows-azure-platform2/windows-azure-development-142014

Vanleenhove, M. (2011, January 11). *IBM Cognos 10 : review & analysis of new functionality*. Retrieved August 24, 2012, from http://www.element61.be/: http://www.element61.be/e/resourc-detail.asp?ResourceId=181

Warren, T. (2010, January 9). *Microsoft Office is US #1 selling software product with one copy sold every second*. Retrieved August 27, 2012, from www.winrumors.com: http://www.winrumors.com/microsoft-office-is-us-1-selling-software-product-with-one-copy-sold-every-second/

Wikipedia. (13, August 2012). *Agile software development*. Retrieved August 2012, 6, from Wikipedia: http://en.wikipedia.org/wiki/Agile_software_development#cite_ref-Agile_Manifesto_0-0

Wikipedia. (2012, August 24). *Business intelligence*. Retrieved August 24, 2012, from http://en.wikipedia.org: http://en.wikipedia.org/wiki/Business_intelligence#cite_note-2

Wikipedia. (2012). *Cognos*. Retrieved 08 26, 2012, from www.Wikipedia.org: http://en.wikipedia.org/wiki/Cognos

Wikipedia. (2012). *Microsoft SQL Server*. Retrieved August 12, 2012, from www.Wikipedia.com: http://en.wikipedia.org/wiki/Microsoft_SQL_Server

Wikipedia. (2012). *SAP NetWeaver Business Intelligence*. Retrieved June 15, 2012, from Wikipedia: http://en.wikipedia.org/wiki/SAP_Business_Information_Warehouse

Wikipedia. (2012, September 20). *Table_(database)*. Retrieved September 23, 2012, from http://en.wikipedia.org: http://en.wikipedia.org/wiki/Table_(database)

Wikipedia. (2012, August 10). *Unified Process*. Retrieved August 16, 2012, from Wikipedia: http://en.wikipedia.org/wiki/Unified_Process

Wikipedia. (2012, August 23). *Waterfall model*. Retrieved August 24, 2012, from Wikipedia: http://en.wikipedia.org/wiki/Waterfall_model

Wikipedia. (n.d.). *Wikipedia*. Retrieved August 15, 2012, from Wikipedia: http://en.wikipedia.org/wiki/Software_development_methodology

APEENDIX A: Engine Domain Execution Sequence

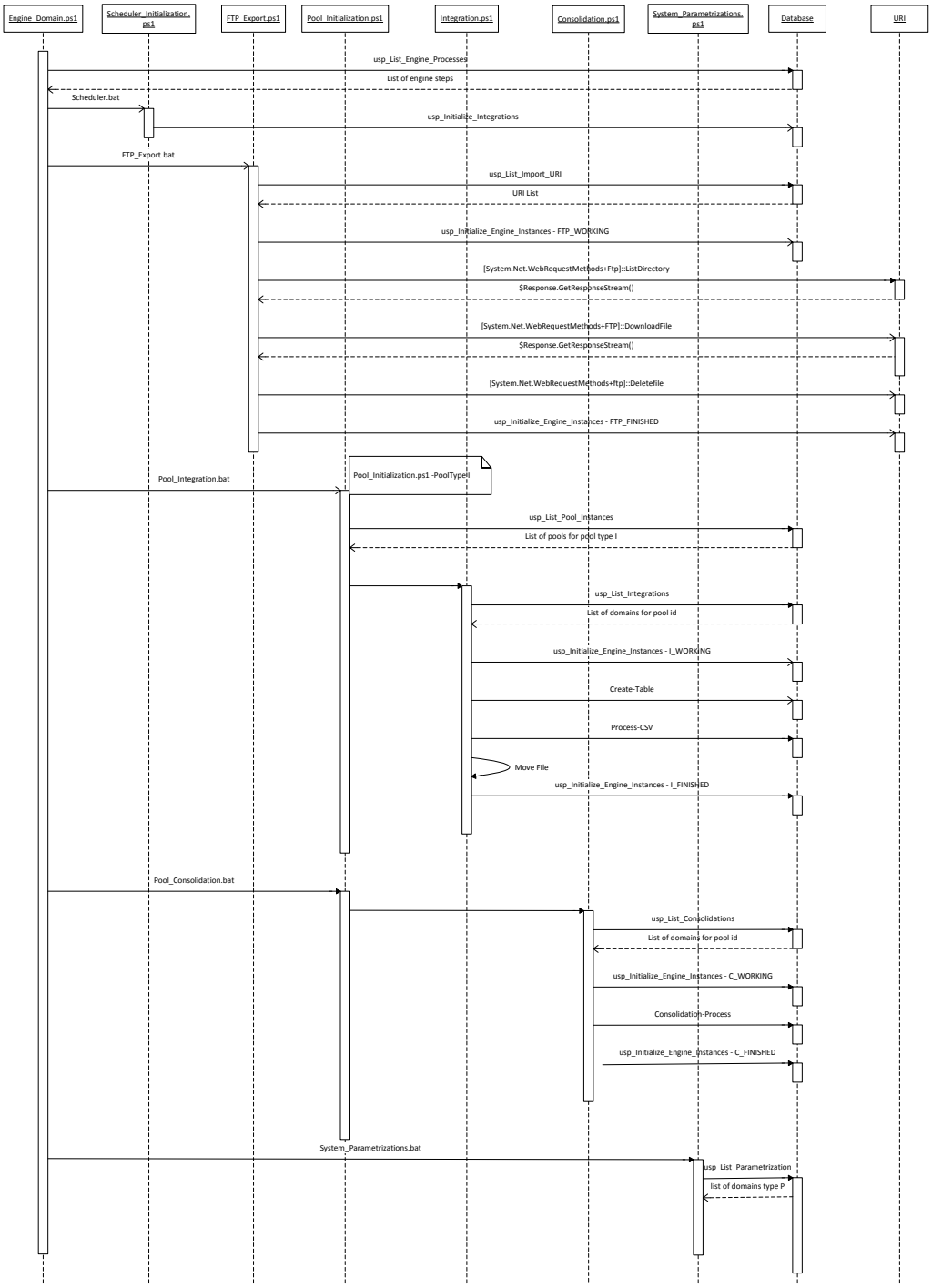

Figure 45 - Engine Domain Execution Sequence

APPENDIX B: Development Best Practices

“Due to the lack of widely-accepted guidelines for database development best practices, it is exceedingly difficult for many IT organizations to construct highly optimized and maintainable software. Let’s be clear: It is not enough to create applications that seem ‘good enough for now’. Instead, we need to figure out how to build applications that will work well today, but also be maintainable ten years from now.” (Feuerstein, 2010)

This sentence serves as a guideline. One that developers should not forget, development best practices are essential for the correct and unified development of any software product. This is even truer if the project requires the work of several people from different teams, areas and work experience.

In order to make all developer work with the same way a series of Best Practices were implemented. They were implemented in the following areas:

- General
- Schemas
- Tables & Views
- Functions
- Procedures

General

- Every code alteration must have a new version and alteration description
- When working with dates, use getutcdate()
- When trying to update a record on a table that has the column [LAST_UPDATE_DATETIME], don't forget to use [LAST_UPDATE_DATETIME] = getutcdate()
- When an object (table) has a life cycle, then it also must have [WORKFLOW] attribute. If not, the attribute [STATUS]
- [STATUS] values can be 1 (active), 0 (inactive), -1 (deleted)
- Use delete Top(1000) when big delete where statements are required
- Never make untested changes to a production database

Schemas

There are several schemas for each different area, use the correct one for each new table.

Table 13 - Database Existing Schemas

Schema	Meaning
CONFIG	General Configurations
ICOMMON	General Operational Information
IDS	Integration Data Storage
IMOBILE	Mobile Application Configurator
ISM	IN Store Monitor Operational Data Storage
LOG	Error and Logging Information
IDOC	Feeds Operational Data Storage
DATA	Data Warehouse
REPORT	External Reporting Tools Access

Tables & Views

- Build clustered index on the surrogate key for each dimension table
- Use table suffixes and prefixes

Table 14 - Table Prefixes

Prefix	Meaning
D_	Dimension
F_	Fact
APP_	Application
REL_	Relation
VW_	View

- Use consistent column name suffixes and prefixes.

Table 15 - Column Suffixes and Prefixes

Column	Meaning	Data Type
IS_	Yes/No question	INT
CODE	Referential code	NVARCHAR(25)
DESCRIPTION	Object Short description	NVARCHAR(100)
LONG_DESCRIPTION	Object long description	NVARCHAR(MAX)
STATUS	Object Active/Inactive	INT
REGISTER_DATETIME	When record was created	DATETIME
LAST_UPDATE_DATETIME	When record was last updated	DATETIME

- Always have a Primary Key
- When applicable, the prefix indicates the functionality (eg “engine_”; “sls_ - sales”)

Functions

- Function always start with “fn_”
- INPUT: @p(description). Eg. @pSessionGuid
- VARIABLES: @v(description). Eg @vUserID

Procedures

- All procedures must be properly indented and commented.
- All procedures must start with “usp_”
- All CRUD procedures must respect the standard nomenclature identifying the procedure type by “CREATE_;UPDATE_; DELETE_; READ_”
- All Upper Tier Procedures must use standard nomenclature like “Get_; List_” identifying the procedure outcome.
- All procedures must have a small header description like presented in the following table

Table 16 - Code Header Description

```
/*****************************************************************************
**********
Module      : usp_Get_ISM_Metrics_HUB
Created By   :  Frederico Couceiro
Created On  : Ago 04, 2012
Description : This usp is a hub for all usp used to return data to ISM
@pFilters
   <filters startDate="05/09/2012" endDate="15/09/2012">
         <list type="ORGANIZATION">
                  <item id="3" />
                  <item id="4" />
         </list>
         <list type="MERCHANDISE">
                  <item id="35254" />
                  <item id="4234" />
                  <item id="563" />
         </list>
         <list type="EQUIPMENT">
                  <item id="55" />
         </list>
         <!-- outras listas de entidades -->
```

```
</filters>

@pACTION:    1  -- Total Consumption - Total
                  2 -- Employee Information
*******************************Modification History *******
Who          Date          Description
*****************************************************
FSC          Set 4, 2012     Version 1.0
*****************************************************/
CREATE PROCEDURE [iCommon].[usp_Get_ISM_Metrics_HUB]
(
@pSession_GUID NVARCHAR(50)
,@pAction  NVARCHAR(5)
,@pFilters  XML = NULL
)
```

- Set NOCOUNT ON
    - First line in EVERY procedure
    - Information about of rows affected by a Transact-SQL statement is not returned.
    - Eliminates the sending of DONE_IN_PROC messages to the client for each statement in a stored procedure
    - For stored procedures that contain several statements that do not return much actual data, this can provide a significant performance boost because network traffic is greatly reduced
- TRY CATCH
    - Error handling for Transact-SQL
    - If an error occurs in the TRY block, control is passed to another group of statements that is enclosed in a CATCH block
    - A TRY…CATCH construct catches all execution errors that have a severity higher than 10 that do not close the database connection
- Variables
    - INPUT: @p(description). Eg. @pSessionGuid
    - OUTPUT: @o(description). Eg @oSurveyID
    - VARIABLES: @v(description). Eg @vUserID