Trends in Open Source Assurance

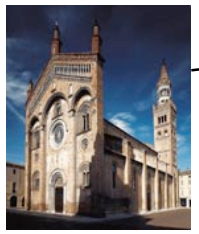
Ernesto Damiani
University of Milan, Italy

OSS 2008 Milan, Sept 8th 2008

Our research lab

- Secure Software Architectures and Knowledge-based systems lab (SESAR)
  http://sesar.dti.unimi.it/
- Part of the Department of Information Technology, University of Milan, Italy

Located on the new campus in Crema, 40 km south-east of Milan



Background

- The open source paradigm is giving rise to new methodologies, competences and processes that need to be investigated both from the technical and the organizational point of view.
  - Companies are increasingly using/bundling OSS within their products.
- Legal perspective: licensing and intellectual property sharing issues
- And many others …

Background (2)

Do we really need OSS research?

Yes:

–Traditional software engineering alone cannot do it

–OSS cannot be tested, secured, certified like traditional software

–Not a problem of licensing, but of driving community-based development and evolution of complex products

Background (3)

- **Multiple viewpoints:**

– *Process-oriented perspective*: emerging new models for cooperative development of new products and ideas

– *Product-oriented perspective*: new software platforms, components and applications, invading markets traditionally held by proprietary products

– *Business perspective*: new ways of creating value

Emerging trends

- **Trend 1: from horizontal platforms (OS, database) to application-level software (business logics)**
  - New OSS is becoming available for business applications such as ERP, CRM and for level 4 services such as VoIP
  - Needs Business Readiness assessment **(more later)**

Emerging trends (2)

Trend 2: OSS development effort shifts from communities of developers to communities of companies

–"Traditional" communities: No programmer can develop and maintain a complex software product alone

–"New" communities: Companies will (or have to) move critical code bases to OSS to share the burden of its evolution

–Need new process quality and discipline without compromising community spirit

Emerging trends (3)

- **Trend 3: OSS users are increasingly companies with a strategy of their own** (rather than individual users willing to save on licenses)
  - These new users find new ways to influence the development community's agenda and methodologies
  - Examples of corporate users "infiltrating" OSS communities, directly or via proxy spin-offs
    - Customers can dictate the pace of technology evolution, both OS-level (multiprocessing, real time) and application level (security)
  - Triggering a change in how OSS is distributed: from SourceForge to OW2

Research and OSS

- Three examples
  - SMP in Carrier Grade Linux (Linux for Telco)
  - Application Level - SWOSS and Business Readiness
  - Process Level: Spago4Q

Example 1: SMP in Linux

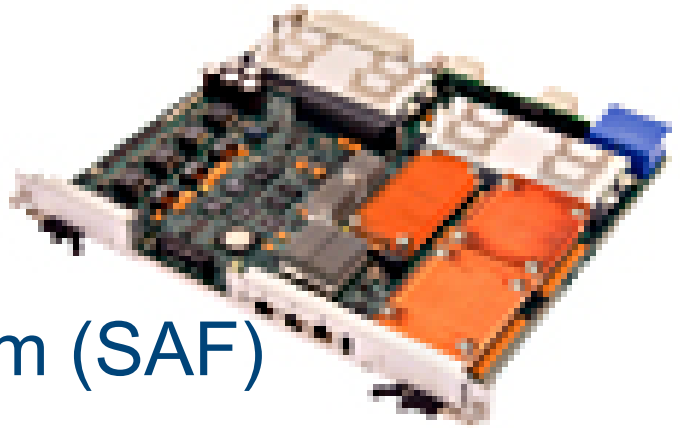
Telco platform requirements

- Enhance application portability
- Simplify deployment
- Meet demands for new data and voice services in fast and flexible way
- Reduce operating costs

- Guarantee service availability of 99,999% or better

Solution

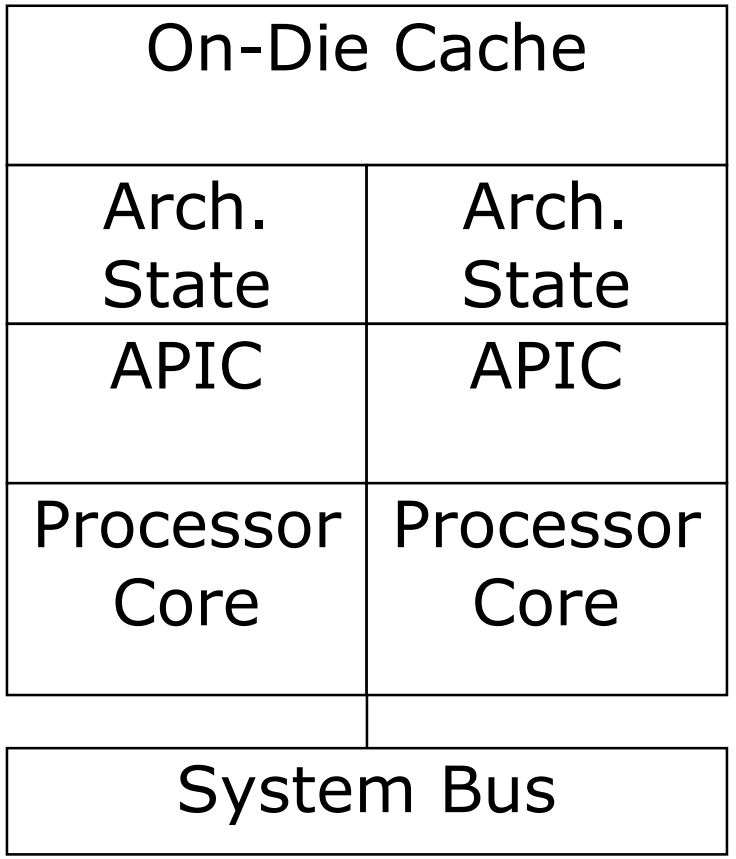
- Hardware: Customer-off-the-shelf
  - General-purpose Symmetric Multi-Processors (SMP)
  - Advanced Telecom Computing Architecture (AdvancedTCA)



- Software: Open-Source
  - Real-time Linux
  - Service Availability Forum (SAF)

Multicore Multiprocessing

- Single main memory for more processors
- Multi-core processors
  - Implement SMP natively in a single CPU
- Moore's Law revisited
  - Number of cores doubles every 18 month



Linux SMP Support

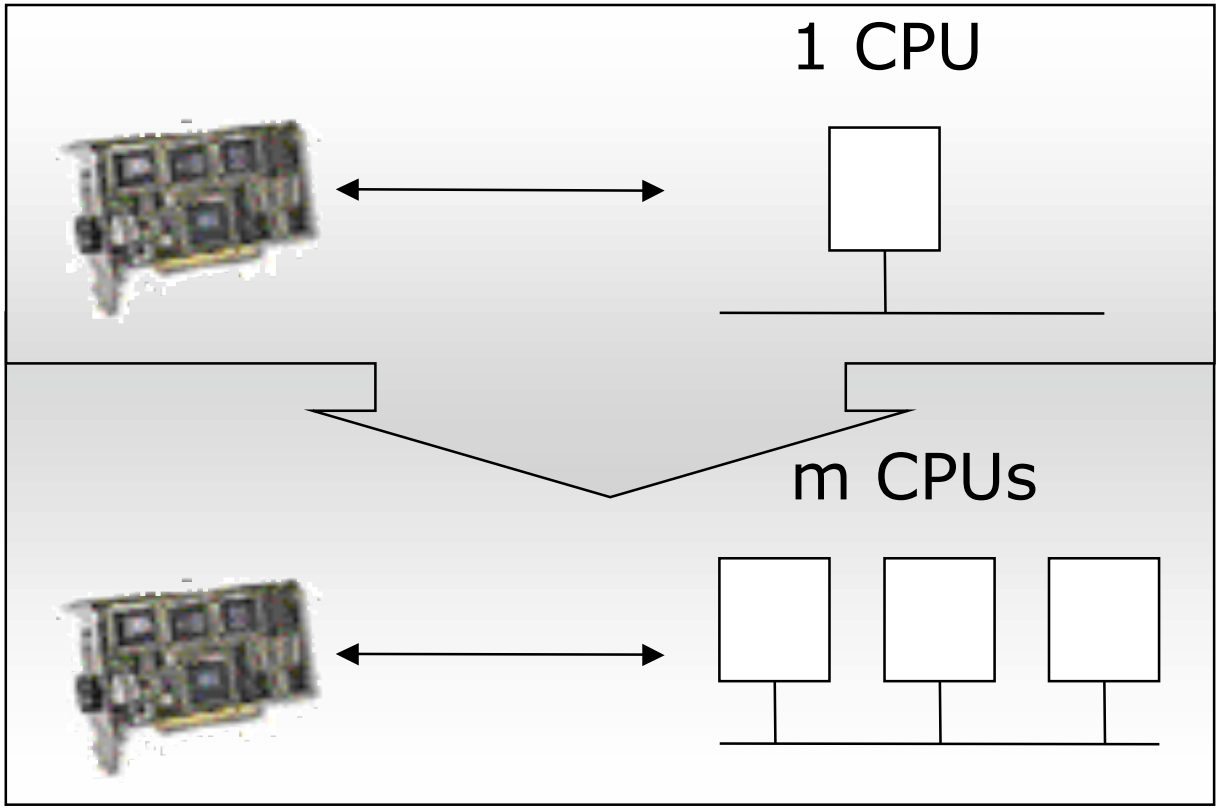
- Since its origin, big effort to bring Linux to SMP
- Most subsystems were compliant with SMP (threaded) since version 2.2 (1999)
  - Synchronization mechanisms on each kernel data structure
  - Process affinity settings
- Software needs to be SMP tailored

Linux SMP problem

- Network stack evolved in 2.4.x and 2.6.x series
- Difficult to adapt further because of dependencies
  - I/O packet flow sequence
  - Packet processing done in kernel in interrupt context
  - Number of executing CPUs is limited to one number of network interrupts

Linux SMP problem

- How to scale SMP with Linux for packet processing with limited network resources?



16

Decomposition techniques

- Application level: Scalability problem solved with application decomposition modeling
  - Technique to parallelize serial application
- Task level: Distinct interrupts balanced over distinct cores
- Data level: Split network flow from same interrupt into partitions; execute on each partition same function on distinct cores

Data Flow Decomposition

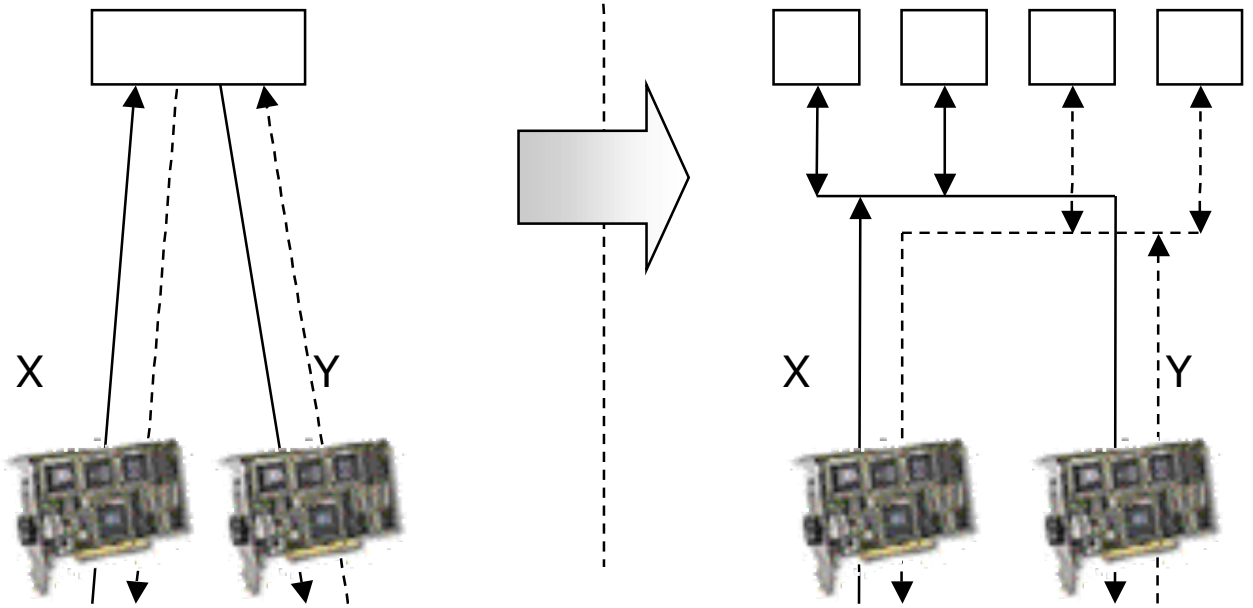
- Packet elaboration function split in different stages
- Each stage is executed by distinct core
- Easier to implement

Fast Parallel Elaboration

- From data level to data flow

CPU4	CPU5	CPU6
CPU1	CPU2	CPU3

CPU4	CPU5	CPU6
CPU1	CPU2	CPU3



Example wrap-up

- Linux evolution driven by user needs, at the pace user want
- BUT: companies had to invest employee time in OSS
  - Other competitors could benefit
  - Does Linux development guarantee the level of **assurance** needed by the companies (e.g. for certification?)

Assurance revisited

– We use the term **assurance** to refer to all activities necessary to provide enough confidence that a software product will satisfy its users' functional and non functional requirements.
– E.g., for security: security standards specify which security requirements a product should satisfy, while assurance standards specify how to collect and provide the evidence that it does.

Can assurance live with OSS ?

- Usually assurance activities are process-related
- Figure below shows security assurance activities mapped on a traditional lifecycle-based process

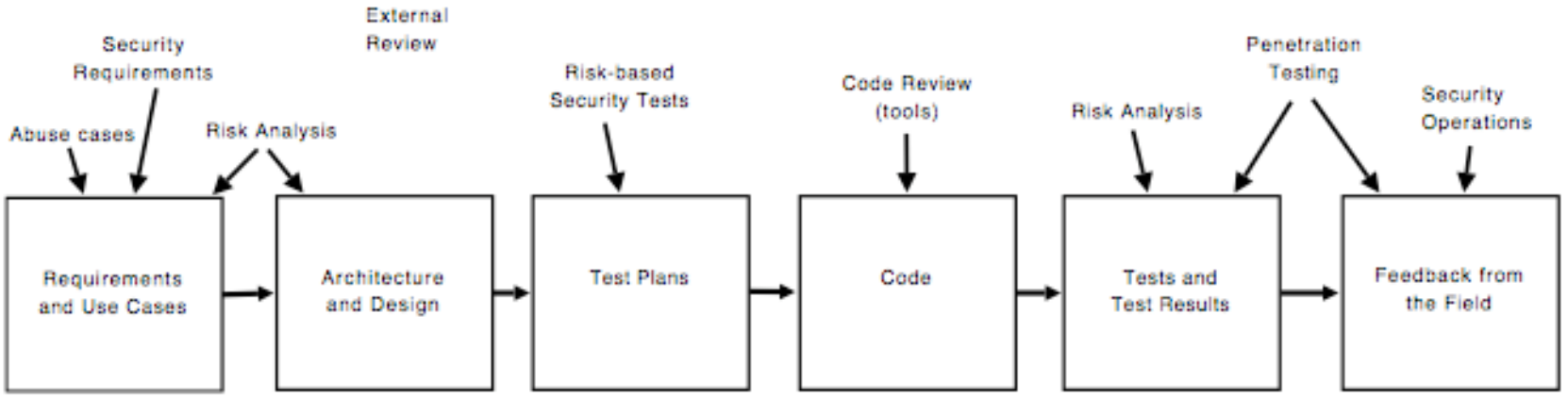


Fig. 5.1: Assurance tasks in a traditional, lifecycle-based development process

Why id OSS development different ?

- Large distributed community of developers
- Rapid release cycles
- Terseness of analysis documentation
- Fast feedback from users
  - Users are an integral part of development process
- Talented and highly motivated developers

It can be done

- Linux two-tiered assurance:

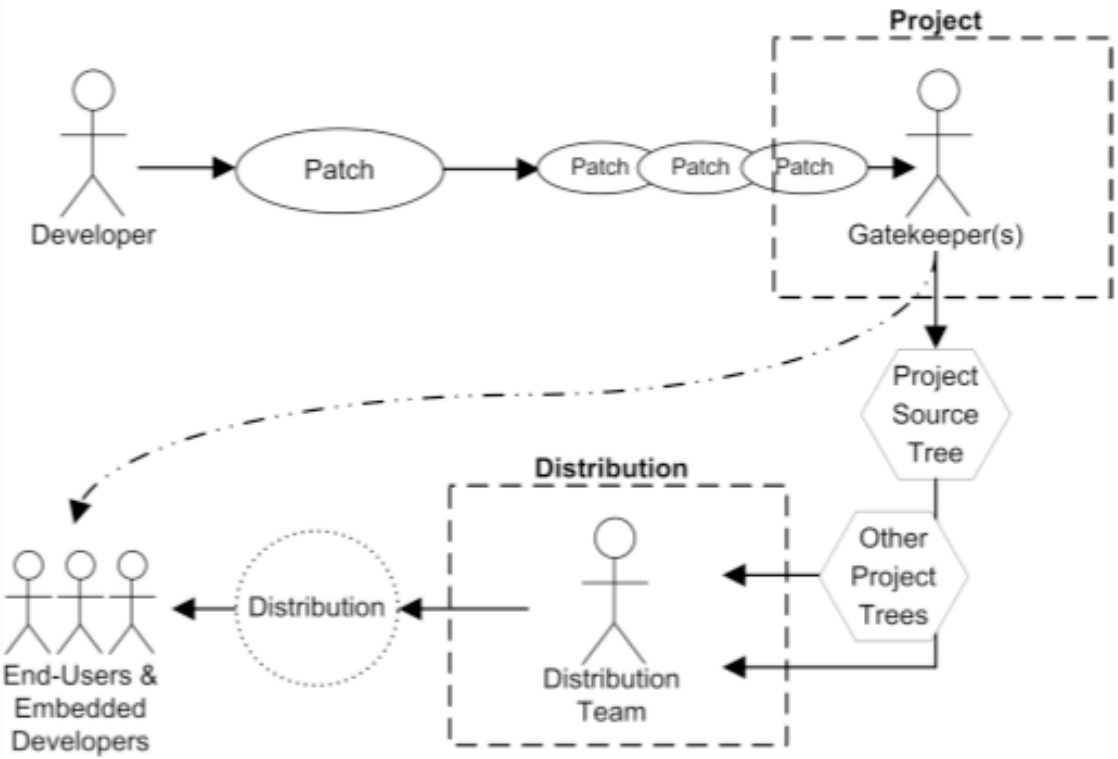


Fig. 5.2: The two- and three-tiered assurance process of OSS

OSS assurance: the role of forges

- Generalized OSS assurance needs agreement:
- Lack of metrics commonly accepted as a reference
  - Lack of published info on reference parameters for OS sw (number of releases, number of core developers, released patches, discussion threads typology, …..)
  - Lack of standard in collecting relevant info (the simple definition of “close” for a bug is different across communities)
  - Lack of common consensus on what is “relevant” (certain communities do not consider relevant the number of downloads, others do)

OSS assurance: the role of forges (2)

- Needs collaboration:
- Metrics and measurements format and semantics
- Division of labor
  - Large communities can agree assurance standards with adopters, but
    - May not be willing to do so
    - one-on-one agreements can prevent standardization
  - Smaller communities just cannot do it
- Forges and competence centers (e.g. Qualipso's) can help in setting up assurance standards
  - Adopters themeselves need to collaborate

Example 2: SWOSS and Business Readiness

The BRR Model

- The Business Readiness Rating (TM) model rates OSS according to its degree of readiness to be adopted within an industrial-strength software system.
- Company-specific adoption guidelines tailor BRR to company needs, focusing on features relevant to mission-critical applications

Spago 4Q
/ * SpagoBI for Quality */

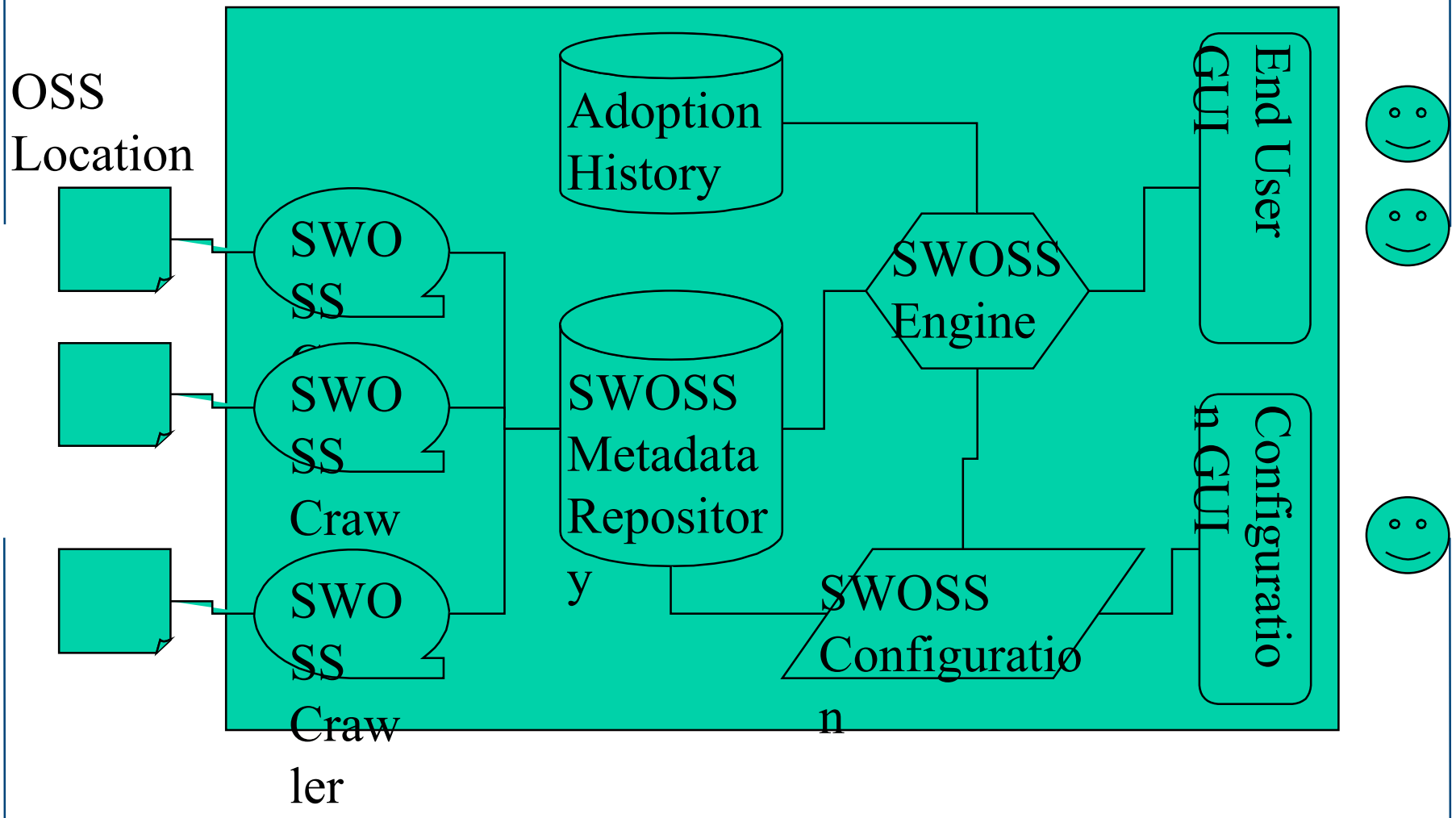
What is new here?

- Again: ASSURANCE
  - Research is needed to enable trustworthy quantitative evaluation of OSS features
- Credible, because of openness
  - Software features can be derived from the code
  - Community claims are not commercials
- However:
  - no standard description metadata
  - no agent-based, semantics aware search
  - analogies with service identification problem

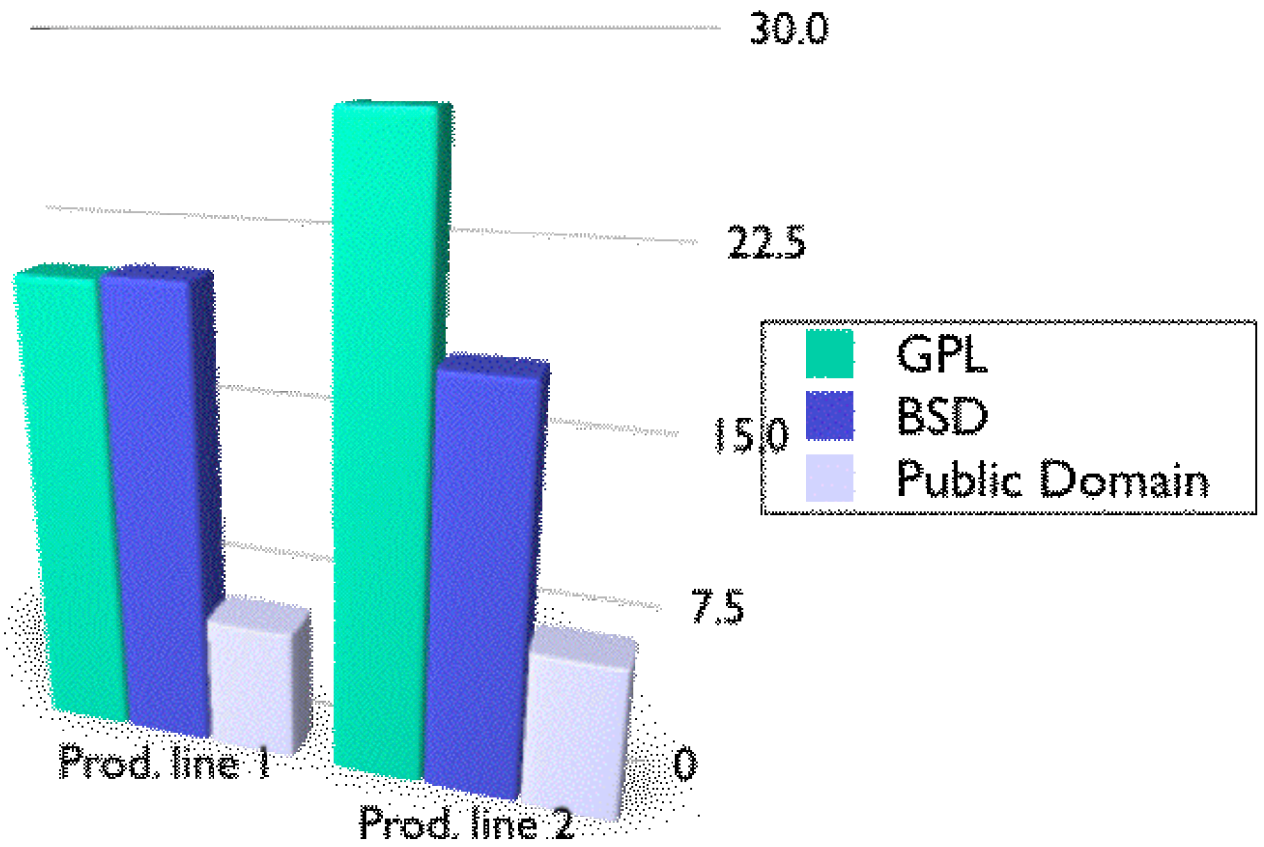
The SWOSS agent

- SWOSS is used with two objectives:
  - Scouting public repositories, such as Sourceforge, to find OS projects that meet a given set of reliability and robustness requirements (readiness).
  - Monitoring private repositories (innersource) to track projects evolution by checking and tracking a specific set of metrics.

SWOSS Architecture Overview



Analyzing products



Operation

- SWOSS Crawlers have two inputs:
  - Keywords and search locations (from user)
    - Include repositories, OSS sites
  - Evaluation parameters
    - Set up at configuration time; based on NSN guidelines
    - A lean object-oriented data model for OSS description
- Crawlers collect metadata and store them in the SWOSS metadata repository.
  - OSS ontology under way

Output

- SWOSS reporting analyzes the metadata and produces a quick and a complete report.
  - Quick report gives a fast feedback on OSS adoptability using a “traffic light” metaphor: green light is a go.
  - Complete report provides a deeper analysis of results covering all parameters foreseen by the model.

Spago 4Q
/ * SpagoBI for Quality */

Example wrap-up

- SWOSS enables and support concrete application of BRR guidelines
- Furthermore:
  - It provides a general description model for OSS, which can be progressively tailored to meet the company needs
  - Supports tracking of adoption decisions and enterprise-wide OSS adoption metrics.

Example 3: SPAGO and Process Monitoring

Developed by

Engineering Ingegneria Informatica

www.eng.it www.spago4q.org

with contributions by

University of Milan - Department of InformationTechnology

SESAR (Software Engineering Software Architecture Research Lab)

http://sesar.dti.unimi.it/

Context

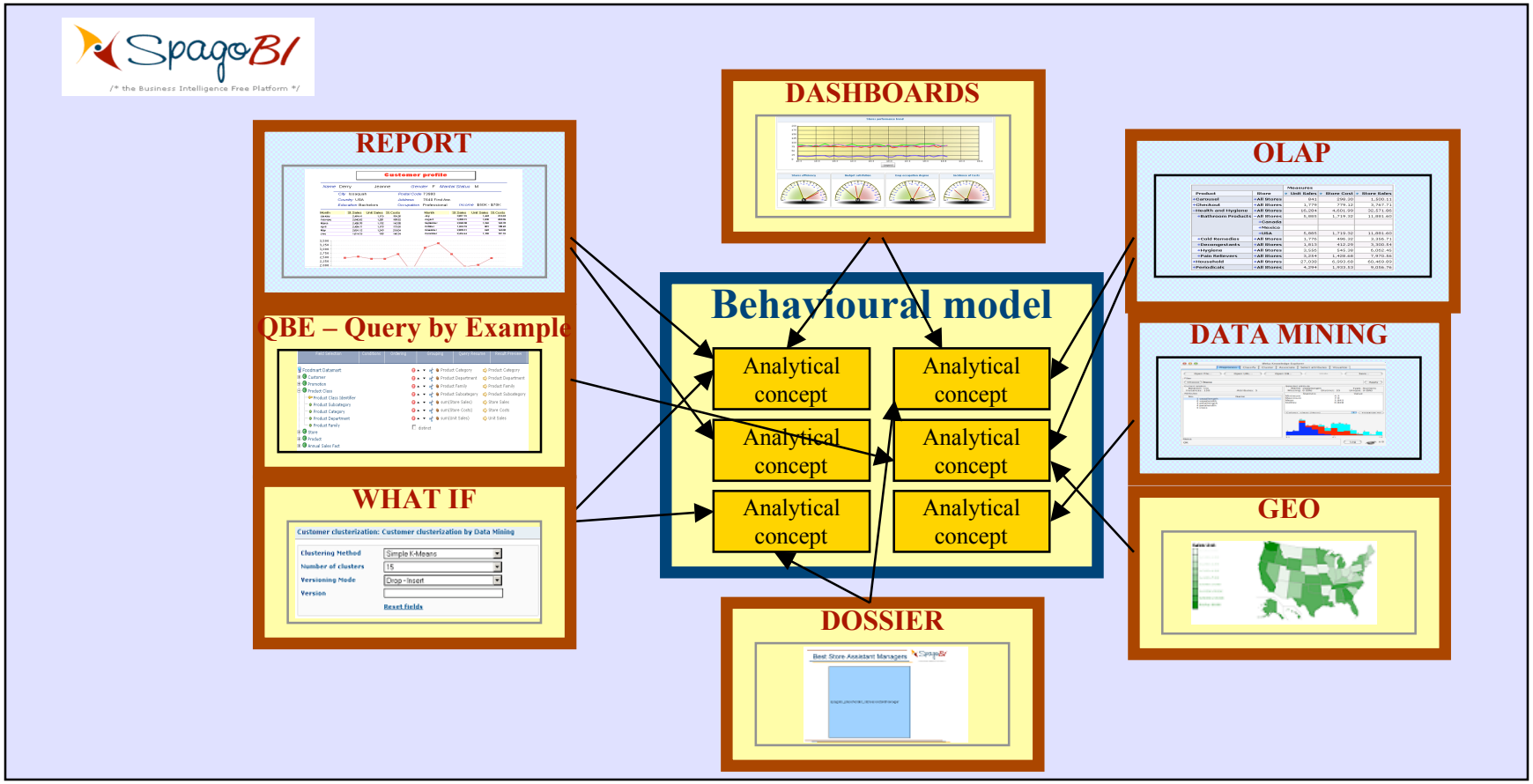
Spago4Q is a FOSS (GNU LGPL) platform for supporting companies and organizations in monitoring process performances in order to improve the overall quality for:

- assessing the maturity of the software development process
- inspecting the quality of the released software

Data and measures are collected from the infrastructure tools with non-invasive techniques.

SpagoBI analytical engines

Spago4Q is a verticalization of SpagoBI www.spagobi.org the Business Intelligence Free Platform.

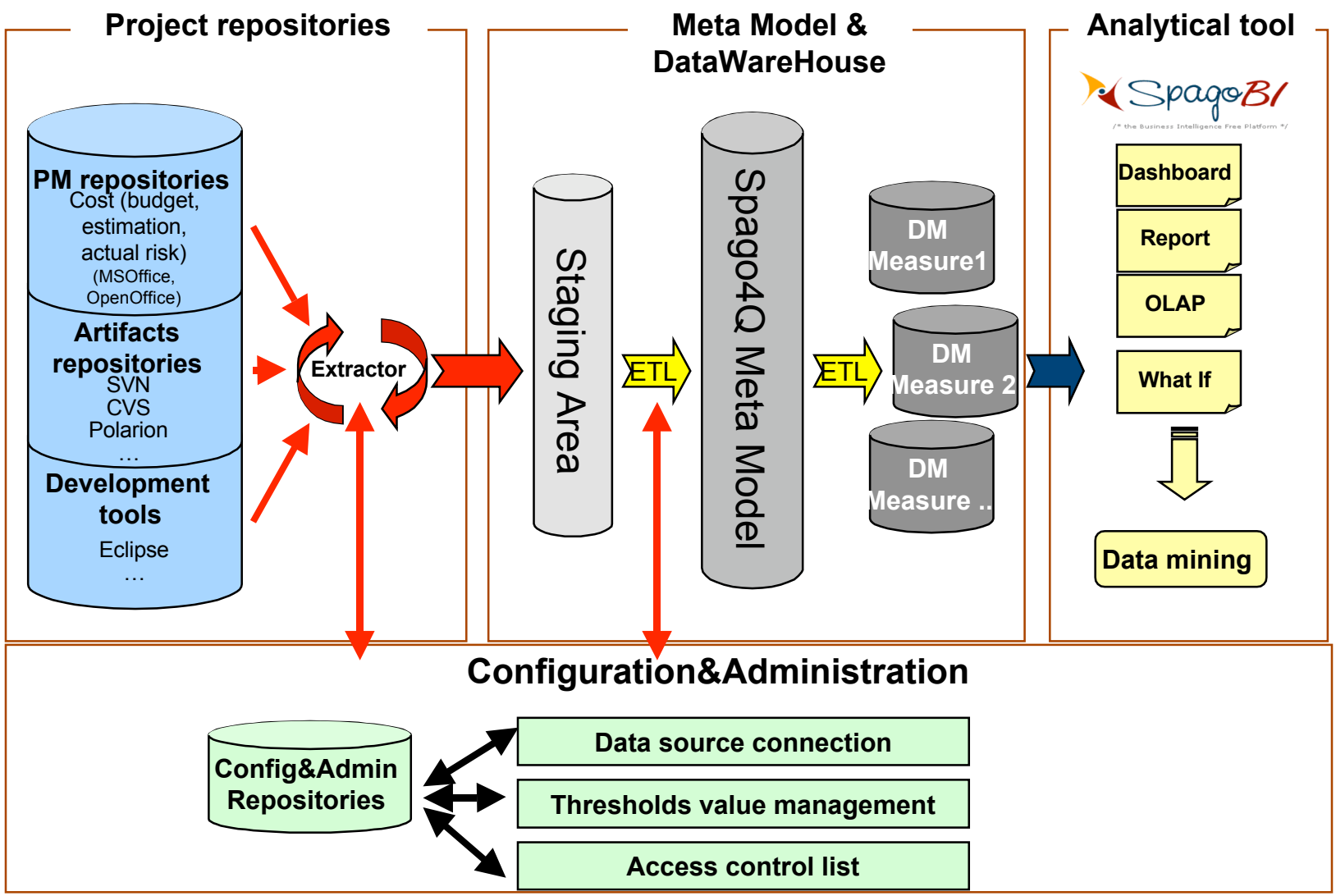


Spago 4Q
/ * SpagoBI for Quality */

Goals

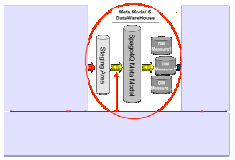
- Highly adaptability to various organizational contexts (imperativeness of the organizational procedures vs flexibility of the company environment)
- Measurement process not bound to the adopted software development process and tools
- Automatic data collection from a set of tools
- Support for a complex system of evaluation
- Measurement's knowledge base: set of "library of measurements" and meta-model instances to satisfy the needs of end users, providing a low cost "out of the box" solution
- Open system and compliance to "de facto" standards

Architectural overview



Main components

- Spago4Q Meta Model 
- Data Warehouse structure 
- SpagoBI portal & analitycal tools 
- Configuration and administration system 



Meta Model

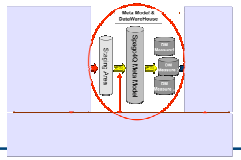
Key Concepts

Abstraction:

- Meta-model adoption for all aspects of process measurement activity
- Consistency between every single instantiated measurement model and the abstract meta-model

Decoupling:

- Software development tools
- Data representation
- Reporting framework

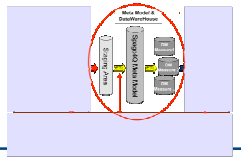


Meta Model

The Spago4Q Meta Model defines:

- The organizational structure (Company/Business Units/Projects)
- The software development methodology (waterfall, evolutionary, UP,SCRUM, etc)
- The measurement framework (GQM model, etc)
- The assessment framework (CMMi, ISO9001-2000, etc

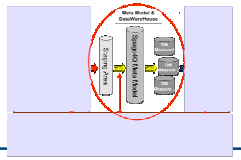
The Spago4Q Meta Model is compliant to MOF (Meta Object Facility) standard.



Meta Model

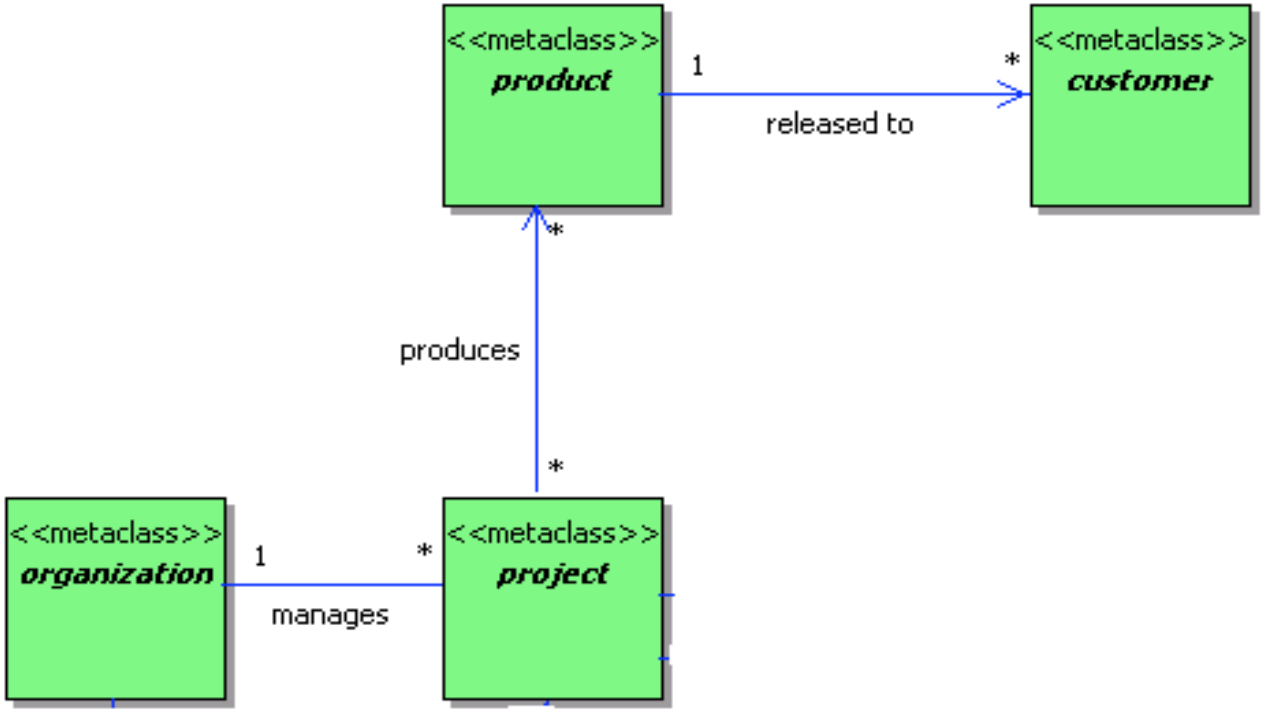
MOF Metadata architecture

META-LEVEL	MOF TERMS	EXAMPLES
M3	meta-metamodel	"MOF Model"
M2	meta-metadata metamodel	UML Metamodel
M1	metadata model	UML Models
M0	data	Modelled systems



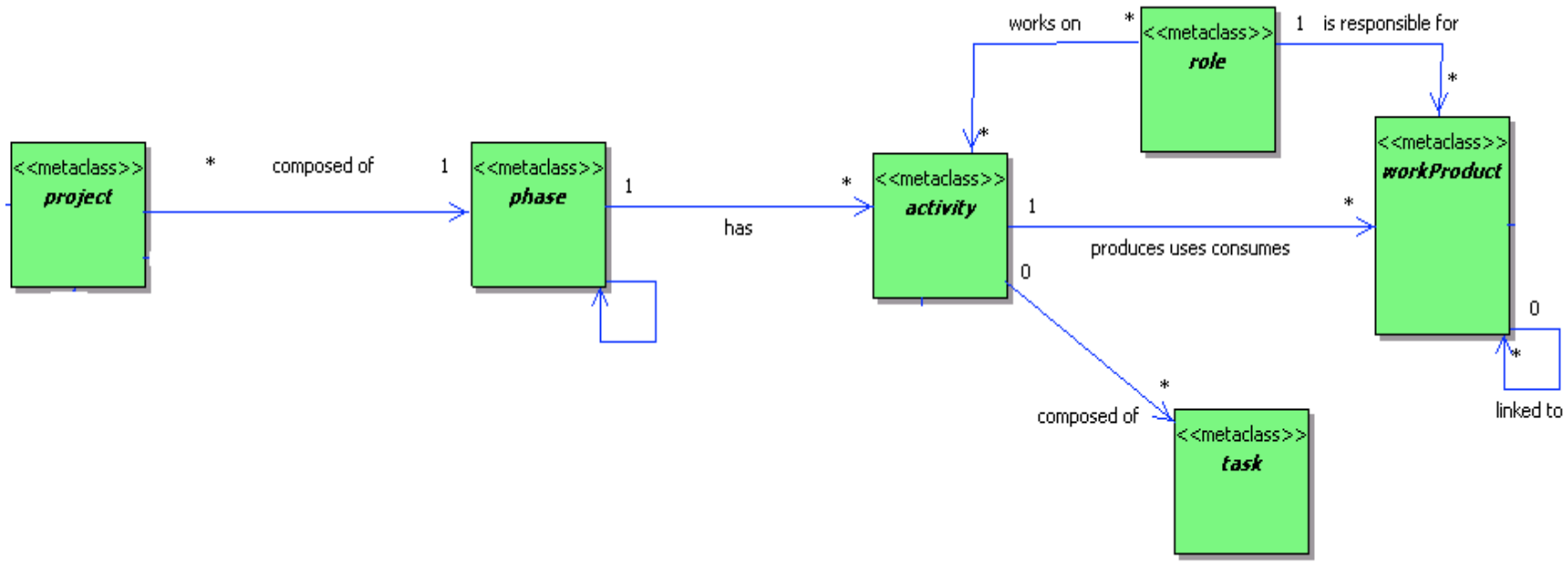
Meta Model

M2 – Organizational structure



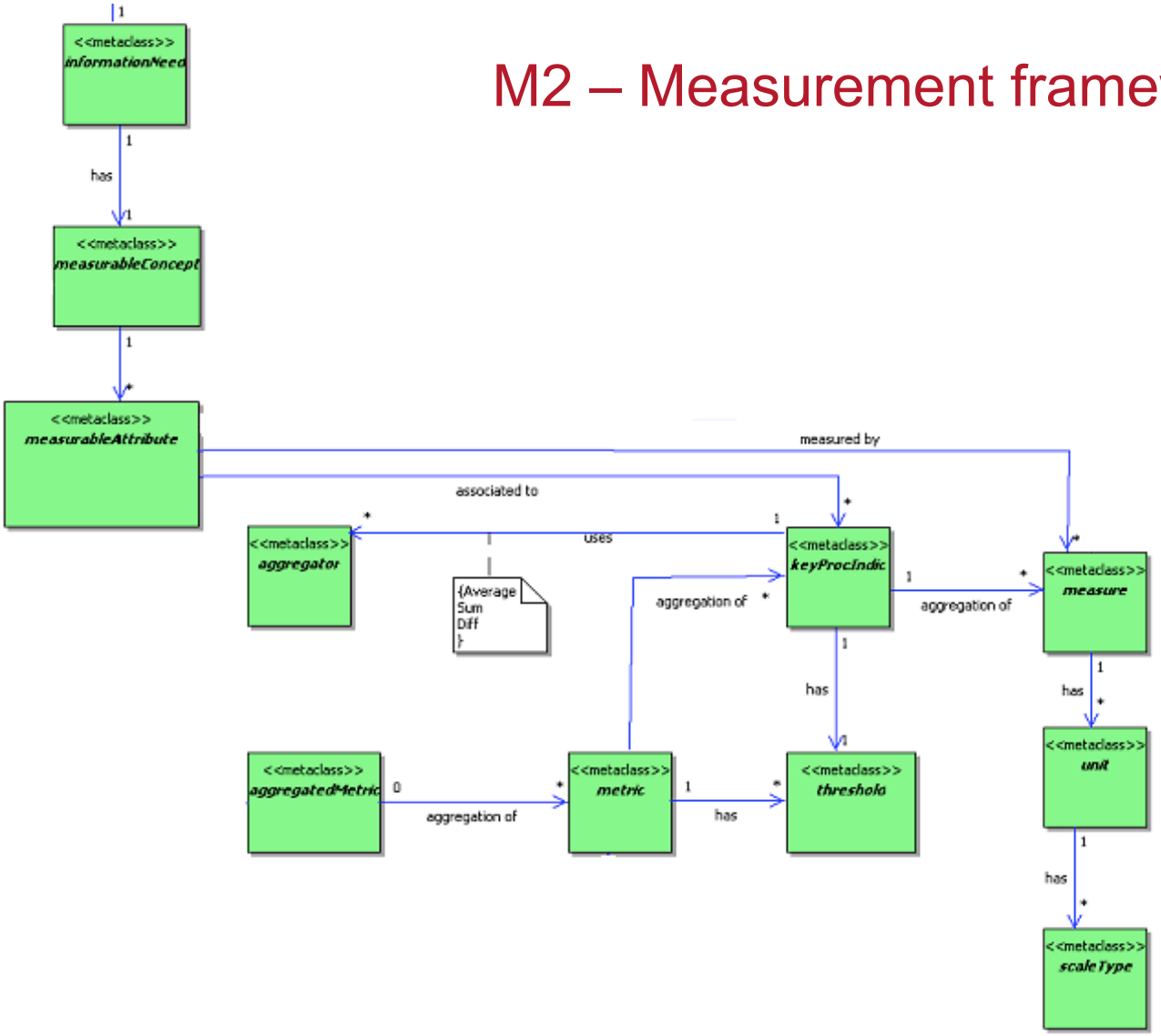
Meta Model

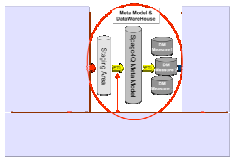
M2 – Development methodology



Meta Model

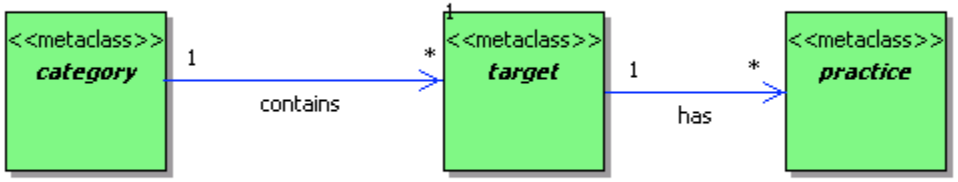
M2 – Measurement framework





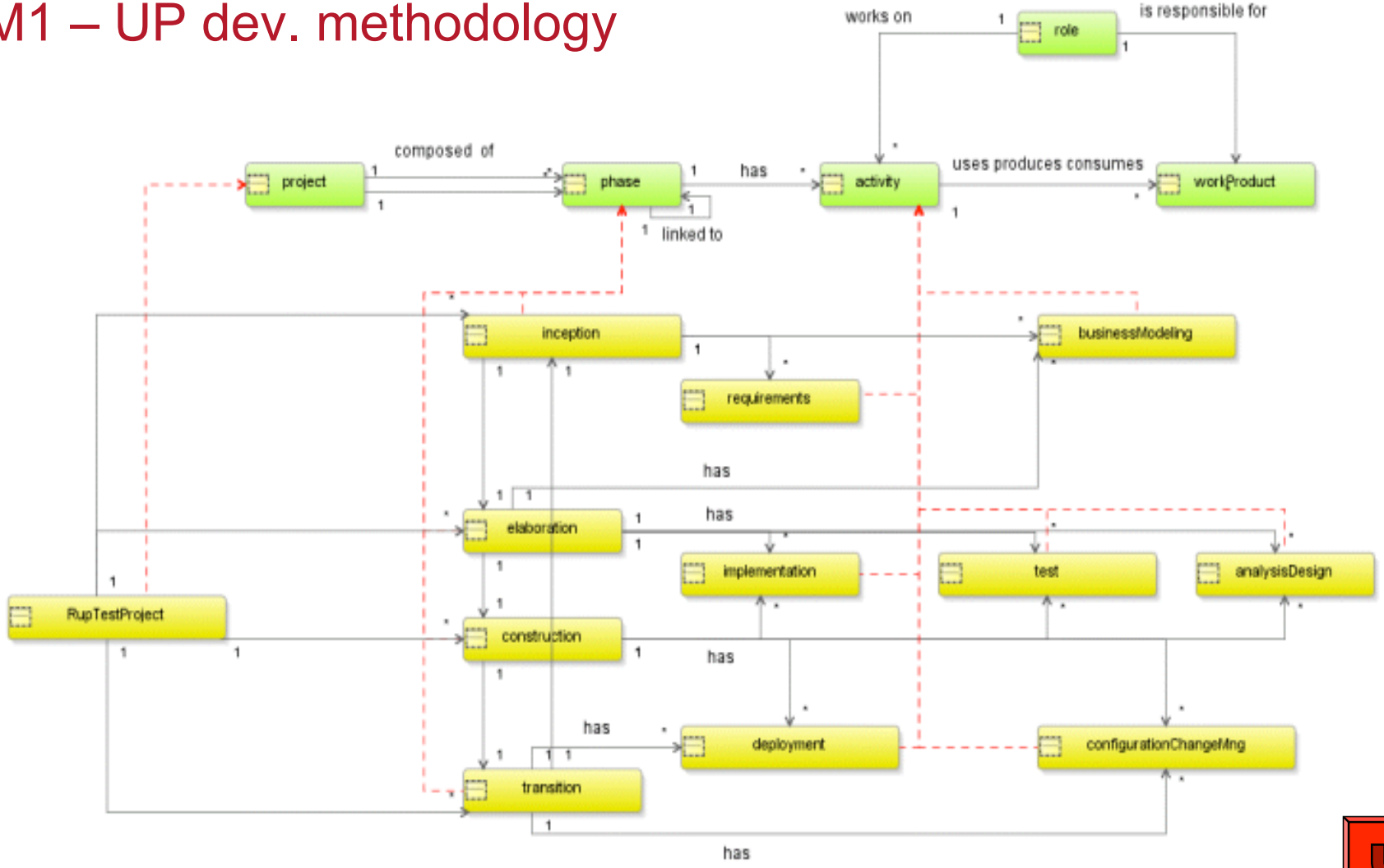
Meta Model

M2 – Assessment framework

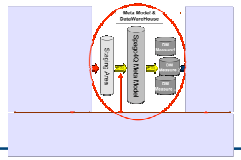


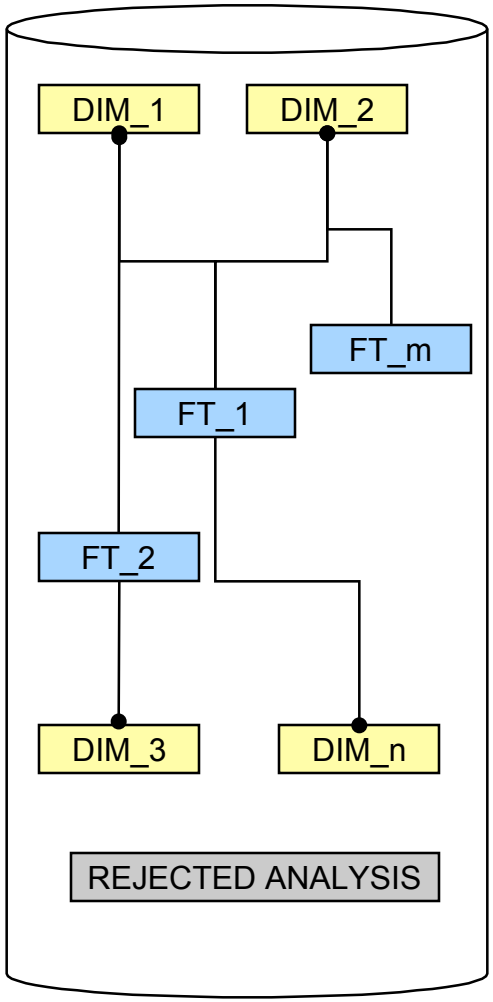
Meta Model

M1 – UP dev. methodology



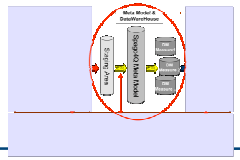
DWH structure





- Snowflake schema
- Fact table: one record for every event occurred on a measurable attribute relevant to a work-product
- Dimension table: conformed dimensions, shared across every work-products
- Historical depth
- Tracking of rejected data

Analytical Tools



Datawarehouse

KPI / Metric / Aggregated Metric and thresholds

DIM_1
DIM_2
FT_m
FT_1
FT_2
DIM_3
DIM_n
REJECTED ANALYSIS

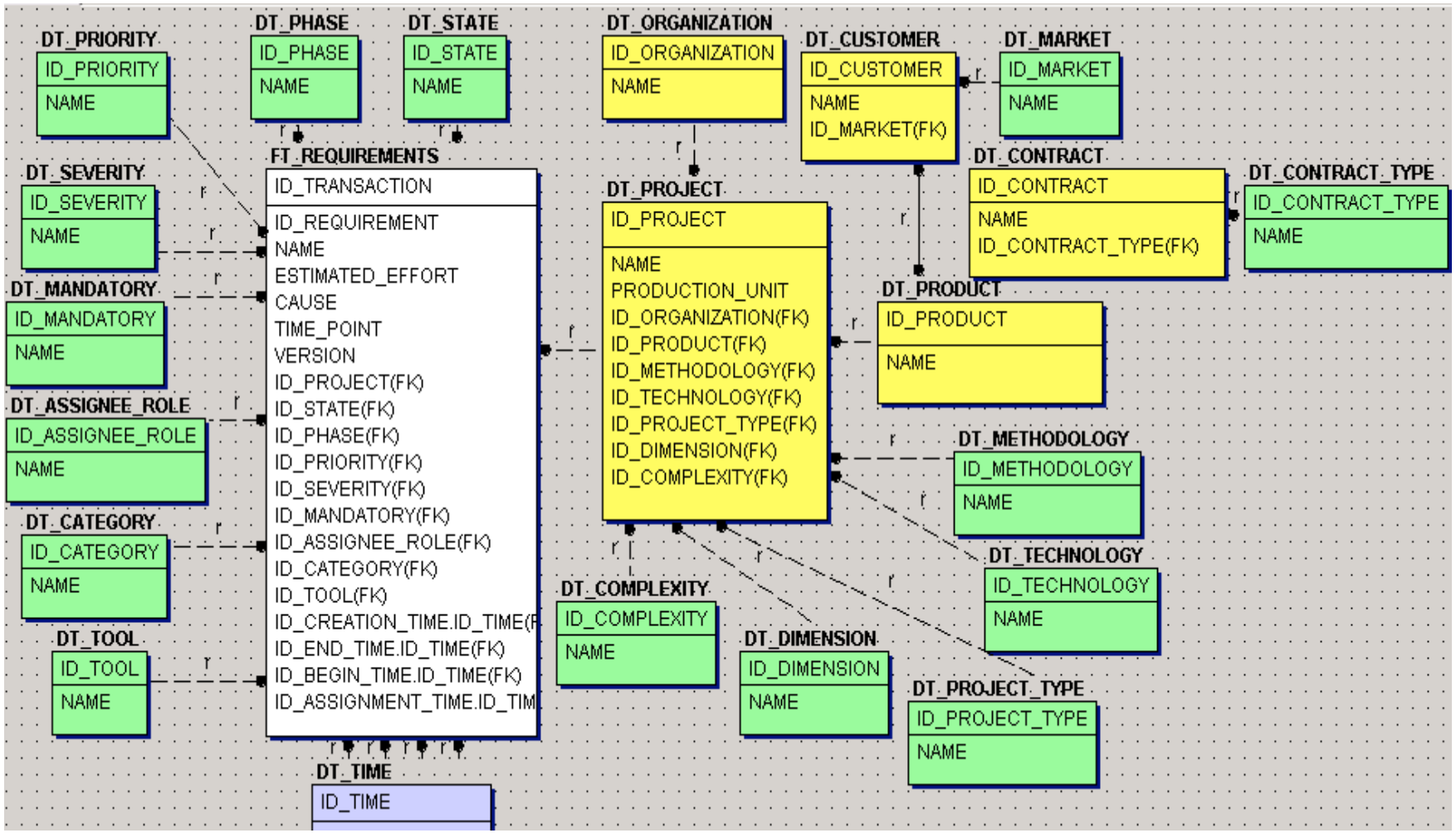
Dashboard
Report
OLAP
QBE
Data Mining

DIM_1
DIM_2
FT_1
DIM_3
DIM_4

Data Mart

(Cross Categories)

Prototype: DWH



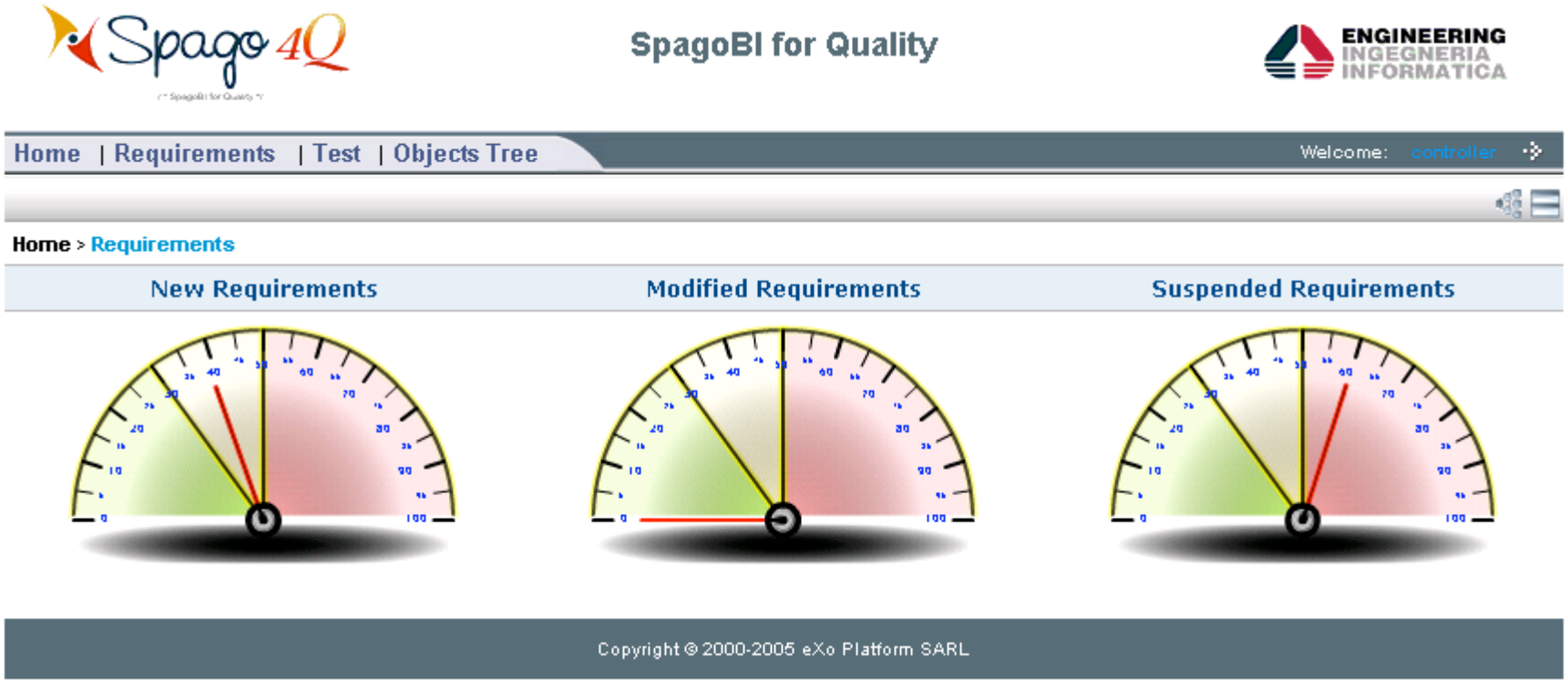
SpagoBI portal

SpagoBI, portal and analitycal tools, representing every KPIs, metrics and the related thresholds as an instance of a analytical document type:

- report,
- OLAP,
- dashboard,
- data mining,
- free enquiry

SpagoBI portal

Example of dashboard: Requirements



Road Map

- A prototype is available to evaluate the capabilities of the platform and its compliance to the requirements.
- The project has been accepted by OW2 consortium (www.ow2.org) visit the dedicated web site (www.spago4q.org).

Example wrap-up

Spago4Q will be driven by a big community of both research projects and industrial projects, helping them to meet their own goals.

Conclusions

- Open Source System Development can act as a lever to lift the burden of innovation
  - Complex system evolution is extremely costly
  - No individual or organization, however big, can do it alone
- Also, it poses new research problems and requires new approach to old one
  - Puts again application within reach of the innovator
  - No need to lawyer up

Some Initiatives

- Forthdoming book on OSS security certification
- Look at my lab web site for
  - References
  - Call for book proposal, Springer book series "Business Applications of OSS"
  - Upcoming journal and workshops announcements
  - Some open source software

Thank you for your attention