

# **Process Engineering Modeling**

**Presented to Toronto SPIN**

**Wednesday, October 29, 2003**

**By Vivienne Suen, Osellus Inc.**

**[vivienne@osellus.com](mailto:vivienne@osellus.com)**

- What is a process?
- Why process models?
- SPEM – the highlights
- Process Automation
- The Future

- Time-Cost-Quality trade-off
- Delivering on-time and under-budget
- “Twin-headed beast” of ignorance and haste [McBreen]
- Dealing with knowledge workers – many intangibles
- Newness of the industry
  - Wealth of knowledge is decades, not centuries old
  - Every factor has changed dramatically in the past 30 years: technology, applications, workforce, usage, etc.
- Software systems are becoming more complex, time to market is shrinking – development is becoming increasingly complex and often fragmented

# Unique Complexity of Software Development Endeavours



osellus

- Cannot leverage work/knowledge from other disciplines (e.g. business process)
- Multiple realistic feedback paths
- Changes tend to be adopted in crisis, and are generally misguided
- Complex decision criteria to determine next step – still requires a lot of human interaction and subjective judgment
- Iteration loops
- Non-linear relationships – effects of an action are not proportional to the original action effect. Consider that productivity gained through overtime is not proportional to that of regular time.
- Non-quantifiable components: team motivation, developer exhaustion, organization/project characteristics and context
- Context sensitivity

# What is a “process”?

- Defines *who* is doing *what*, *when* and *how* [Jacobson, Booch, Rumbaugh]
- Different ways of organizing people and resources
- Not just about workflow – a good process should fully define roles, qualifications, and artifacts

- Methodologies optimize specific things:
- Waterfall: efficiency through specialization. Fears unreadability of code.
- Unified Process: correctness and traceability. Fears lack of documentation, wants an audit trail.
- Ad hoc: minimize time and cost.
- Game development: predictable delivery of a stable product (reliance on marketing).
- Open Source: unrestricted access to source code, reputation, and community.
- Agile processes: communication, delivery of working code. Fears doing “too much”.

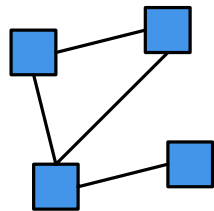
# The Main Process Challenges

- Process Fit
- Process Development and Maintenance
- Can be overwhelming to try to get perfect fit every time
- Ideally, processes should be scalable or easily adjusted to fit individual projects
- Even in the process development dimension, continuous improvement is becoming prohibitive
- Strange thing about the process world is that most processes deal with the creation of new software – while most of the effort in the industry is focused on *maintaining* existing software!

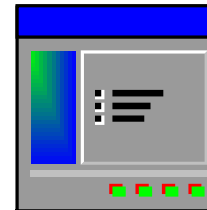
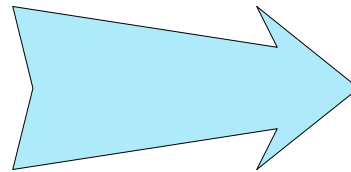
- What is a process?
- **Why process models?**
- SPEM – the highlights
- Process Automation
- The Future

# What is a model?

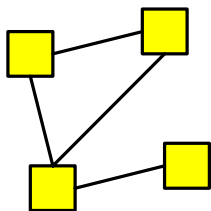
- “Model” = a set of statements describing/prescribing the essential workings of something
- “Process Model” = model of a software development methodology. Describes/prescribes a specific and particular way of developing software.



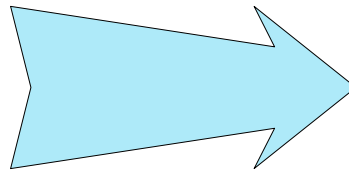
Design Mode



Code



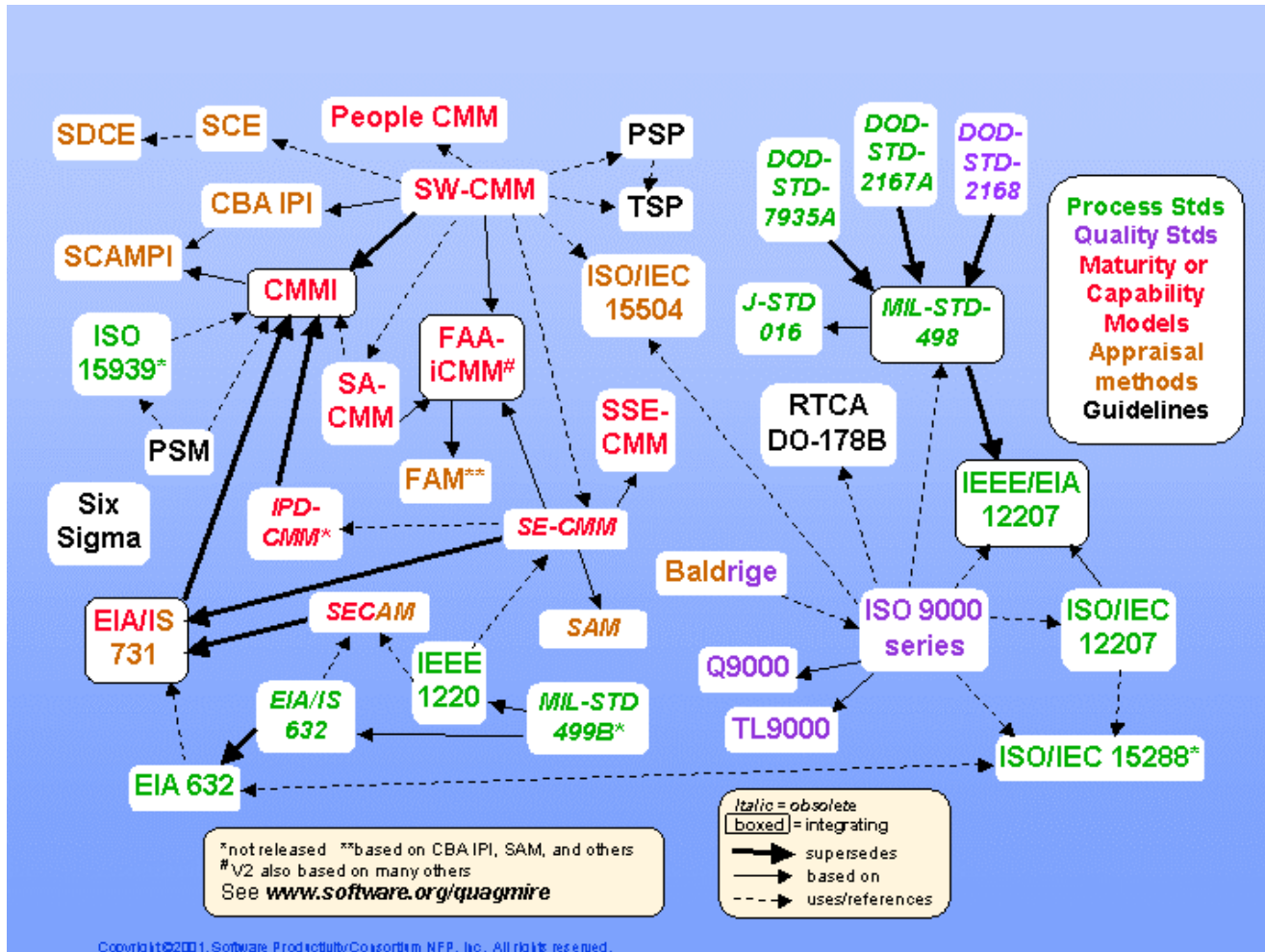
Process Model



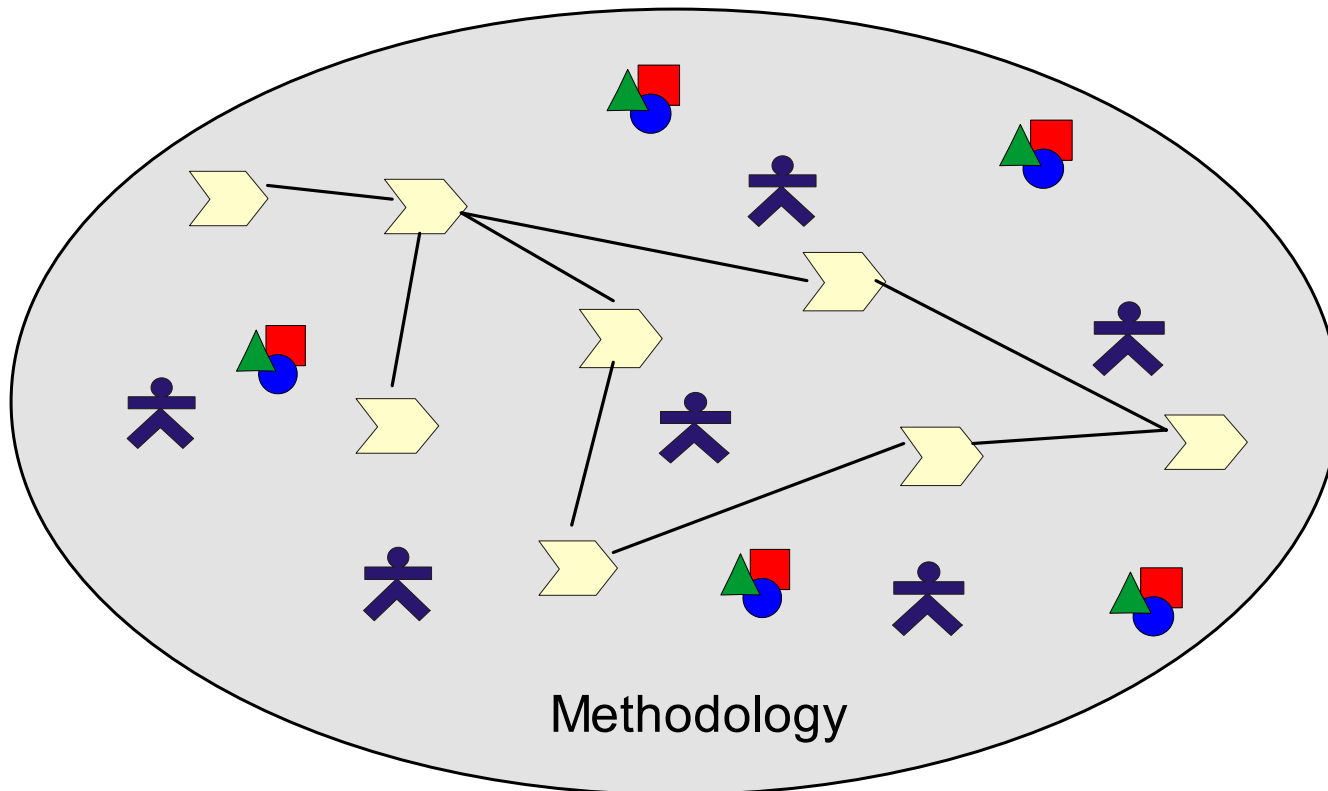
- Why do we model anything?
- Usefulness of models over documents
- Generation of workflows – we are able to reap tremendous gains by generating code from design models, what might we be able to generate from a process model?
- CMM Level 3: Organization Process Definition and Software Product Engineering key process areas essentially mean having a process model
- Codification of best practices: dissemination of methodologies throughout the organization
- Repeatability

- Quality programs
- Better tools
- Offshore/outourcing models
- Change in methodology/process

# Process Modeling – The Current Landscape



- Effort to develop, in a disciplined manner, the roadmap or approach for undertaking a software development project

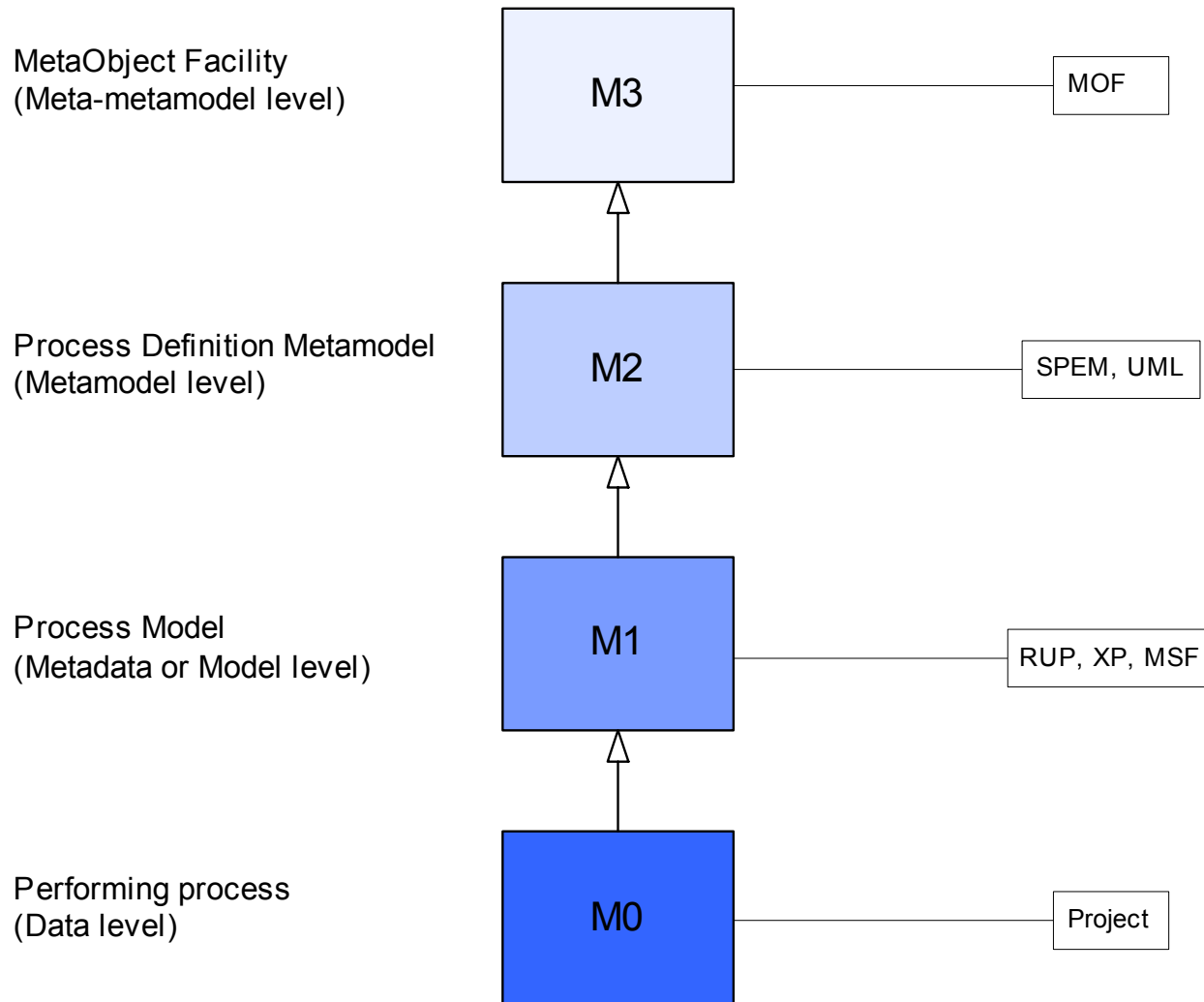


- No common, agreed-upon definition of what software process engineering should comprise
- No standardized approach to designing a process based on requirements
- Numerous and varied drivers – organization and team culture, established practices and habits, project type/size/budget, novelty, cost of failure, process resentment factor, commitment to standards, problem domain ....
- No standard (either open or defacto) for SPE

- What is a process?
- Why process models?
- **SPEM – the highlights**
  - Metamodel Overview
  - WorkProducts
  - ProcessRoles
  - Work Breakdown Structure (workflow)
  - Dependencies
  - Guidances
  - Model Management
- Process Automation
- The Future

- Software Process Engineering Metamodel
- Standardized way of expressing *any* software development process
- Specifically for software development processes
- Vendor, framework, methodology neutral
- Leverages expressiveness and popularity of UML
- [www.omg.org](http://www.omg.org), document number formal/2003-11-14

# Layers of Abstraction



- **Basic Elements** (External Description, Guidance)
- **Dependencies** (Categorizes, Impacts, Import, Precedes, RefersTo, Trace)
- **Process Structure** (WorkProduct, WorkDefinition, Activity, Step, ProcessRole)
- **Process Components** (Package, ProcessComponent, Process, Discipline)
- **Process Lifecycle** (Lifecycle, Phase, Iteration, Precondition, Goal)

- Work Breakdown Structure: modeled with Lifecycles, Phases, Iterations, WorkDefinitions, and Activities.
- Artifacts and roles: modeled with WorkProducts and ProcessRoles, and inputs/outputs and performers/assistants to Activities
- Packages – for modularity and reuse.

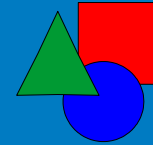
- Process modeling methodologies don't exist ... yet.
- Good approach is to start with concrete elements – WorkProducts (deliverables) and ProcessRoles.

# Break Time!



# Process Modeling Exercise



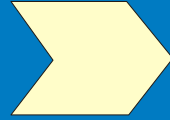


- WorkProducts are the artifacts of a process – any tangible piece of information produced, consumed, or modified
- Could be any format or media – use WorkProductKinds to distinguish
- Can be aggregated, and have state machines
- “Artifact” in RUP and QuadCycle, “Work Product Description” in IBM processes, “deliverable” or “product” in others.



- ProcessRoles are not job descriptions or job titles! A ProcessRole is not a person.
- Meaningful grouping of skills and responsibilities
- Has a parent class, ProcessPerformer.
- “Worker” in UP, “role” in IBM GSM and others, also “agent”.

- The work breakdown structure (WBS) of a process model describes the work to be performed, and the general flow of activities.
- SPEM identifies a set of work breakdown structure elements, at varying levels of detail: Lifecycle, Phase, Iteration, WorkDefinition, Activity.



- Activity: piece of work performed by a single ProcessRole
- 1 performer, any number of assistants
- May consist of atomic Steps, which may be represented by activity graphs
- WorkProducts are the inputs and outputs, via an association class.

# Higher-level WBS Elements

- WorkDefinition: describes a composite set of Activities
- Iteration: composite WorkDefinition with a minor milestone
- Phase: the span of time between two milestones (specific entry and exit criteria). No overlap.
- Lifecycle: describes the behavior of a complete process to be enacted in a particular project (series of Phases)

- Use Preconditions, Goals, and Precedes to construct the work breakdown structure
- Combination of completion-based and condition-based rules is flexible and powerful

- Expressed in terms of WorkProduct states
- Boolean expression, e.g.

Architecture Document == approved && Use Case Model == ready

- Necessary for commencement/conclusion of a work breakdown structure element, but not sufficient!

# Precedes

<<Dependency>> Precedes
kind = finish-start

<<Dependency>> Precedes
kind = start-start

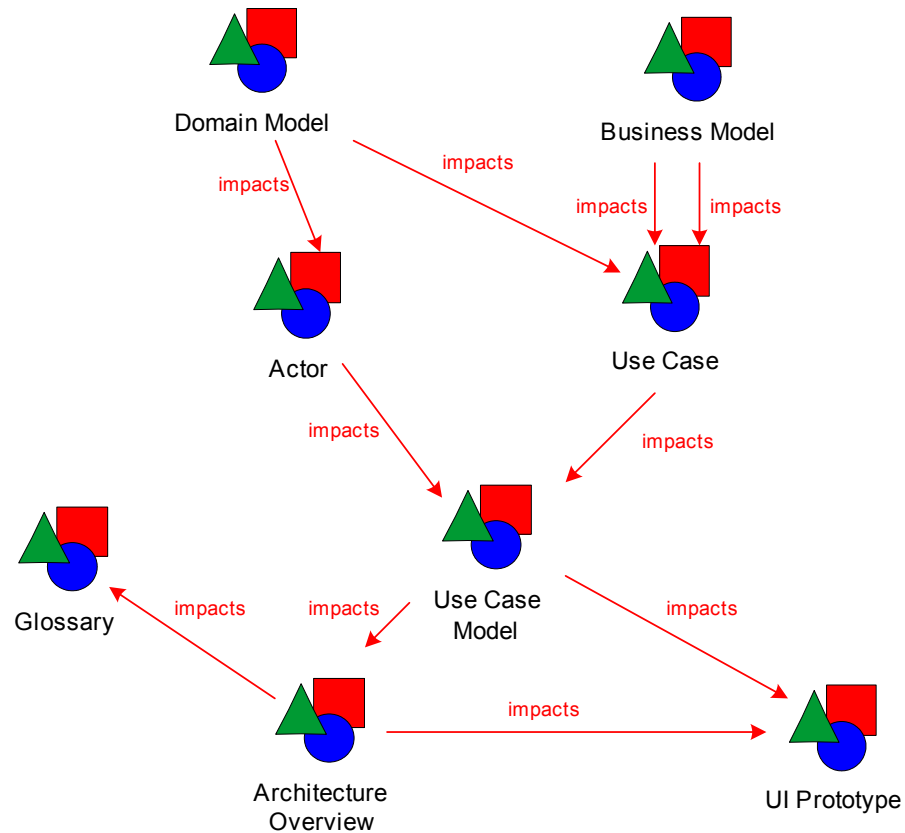
<<Dependency>> Precedes
kind = finish-finish

- Finish-Start, Finish-Finish, Start-Start
- Phases are linked via Finish-Start (since no overlap)



- Impacts Dependency – can use to create a WorkProduct Dependency Diagram
- Trace Dependency – to trace the flow of system requirements through the model
- Guidances
- Disciplines
- Packages

- Can be used to create a WorkProduct Dependency Diagram



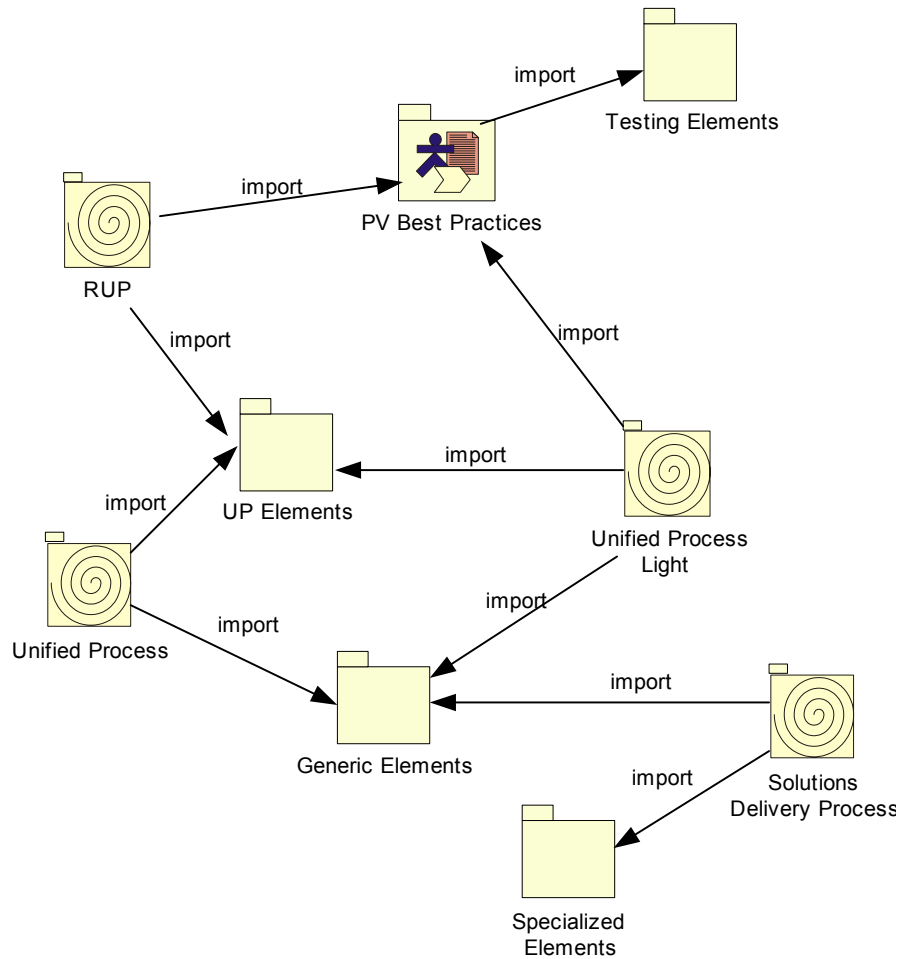
- Are not WorkProducts – neither produced nor modified by the process.
- Any element (or package) may have any number of Guidances.
- Must have an assigned GuidanceKind (Technique, UMLProfile, Checklist, ToolMentor, Guideline, Template, Estimate, Technology Roadmap).

- How do we manage models, once they have been defined at the elemental/structural level?
- Helps ease the path for process customization and evolution

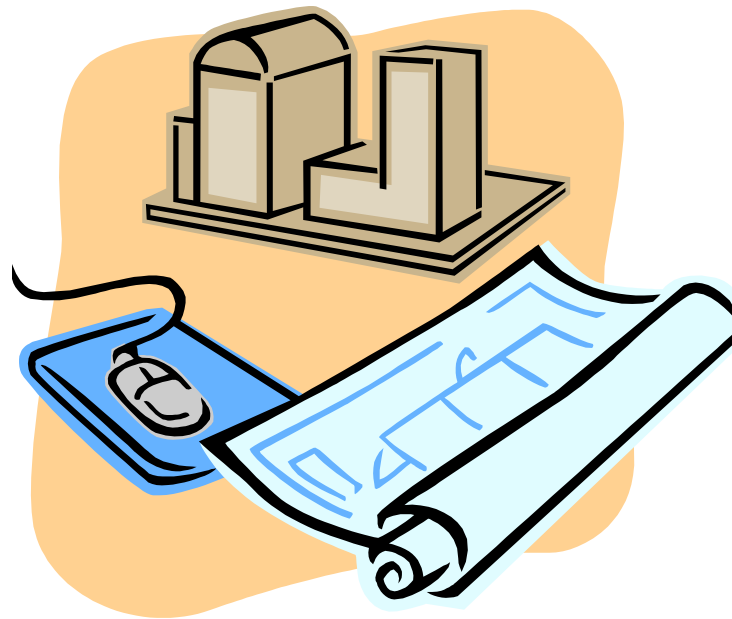
# SPEM Package Types

- Package: as in UML, a container that can both own and import process definition elements.
- ProcessComponent: self-contained, internally consistent piece of process description. May be composed.
- Process: complete, end-to-end process.
- Discipline: specialization of Package used to categorize Activities according to a common theme.

- Modularity and reuse



# Process Model Examples

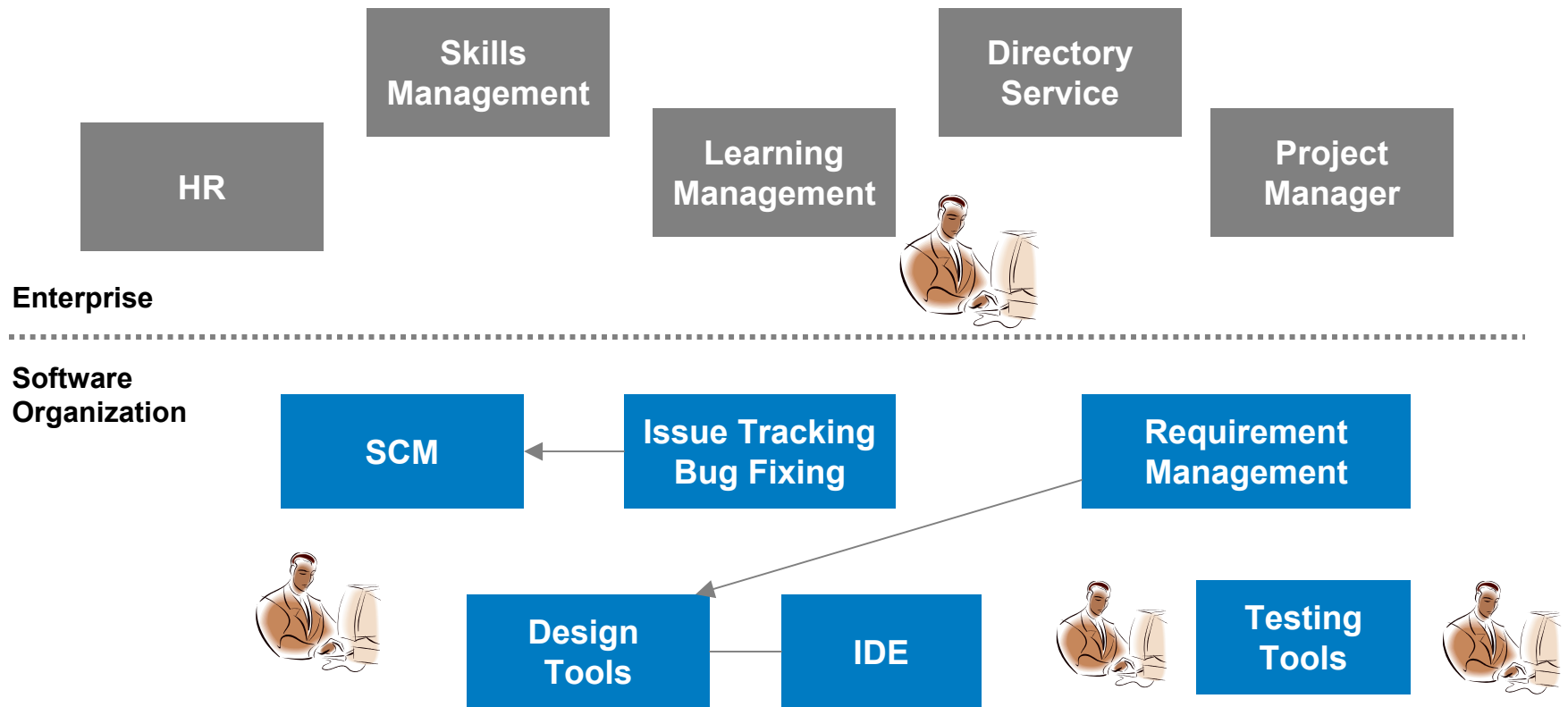


- What is a process?
- Why process models?
- SPEM – the highlights
- **Process Automation**
- The Future

# How to Enforce a Process?

- Training – prioritize based on most heavily-impacted roles (which is easy to discern from a process model)
- Integration of point tools – a well-defined process model may even be tools-aware, and be optimized for the standard developer toolset
- Practitioners play an active role in recommending changes
- Project management also plays an active role in implementing the process itself, and not just the projects.

# Islands of Automation and Integration



# Benefits of automating Software Development Process

 osellus

**Existing process or Best practices**  
(Published or Internally developed)



## Process Automation

Model, Enact and Monitor the chosen process across Software Development



### PMO

Monitoring  
Administration  
Compliance report



### Executives

Management dashboard  
Budgets and performance  
Projects and Risks



### Project Manager

Management by Exception  
Effective Risk and Issue management



### Practitioner

WorkProducts  
Help documents  
Sample cases  
Access to tools  
...and more

- What should we look for/demand in a process automation system for software development?
  - ✓ SPEM compliance
  - ✓ Enterprise foundation
  - ✓ Outsourcing and offshore support
  - ✓ Visual modeling
  - ✓ Skills management
  - ✓ Concurrent enactment engine
  - ✓ Interfaces to point tools
  - ✓ Real-time monitoring

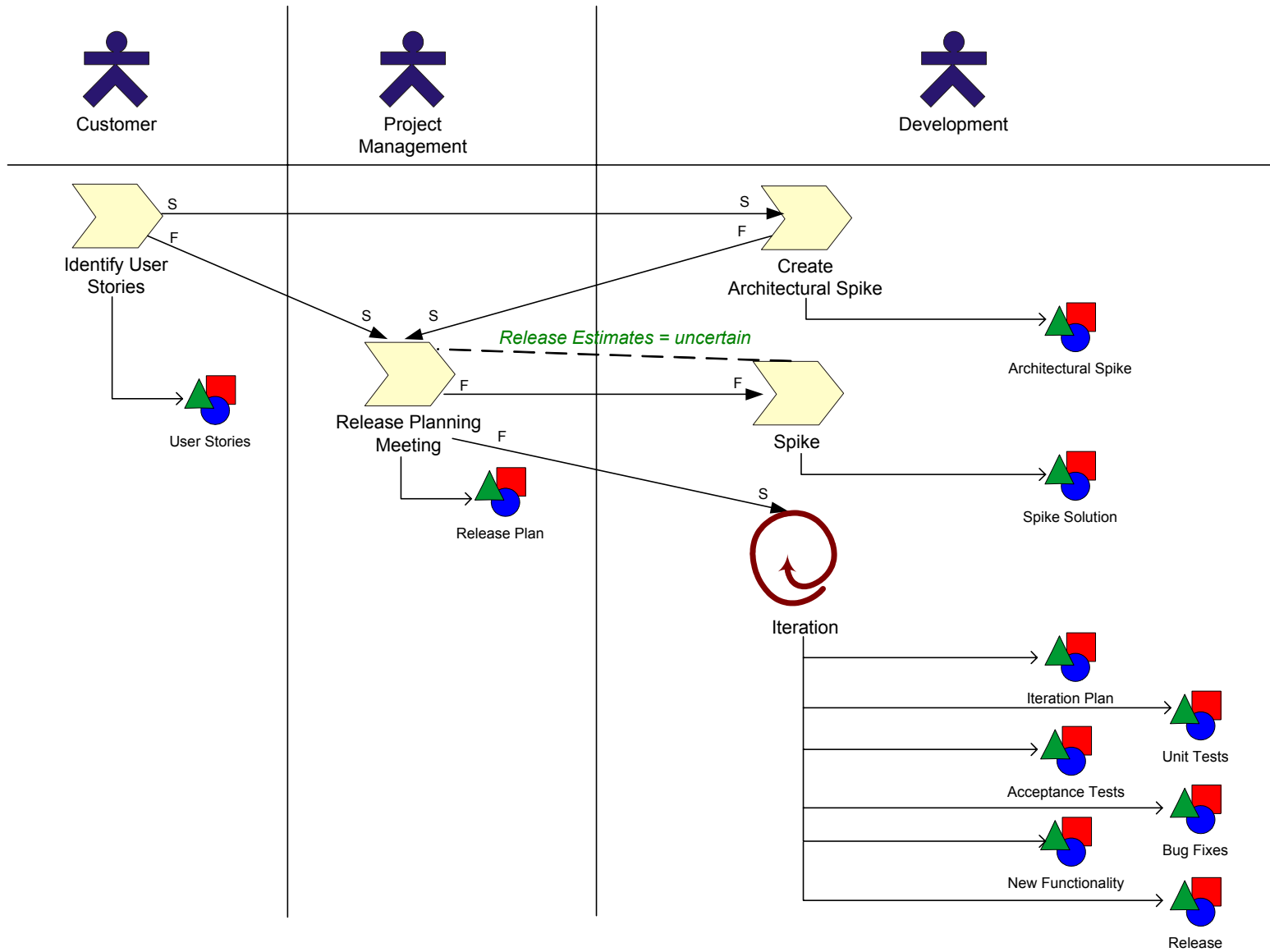
- Monitoring is crucial if you want to learn from your processes
- See how well the process is being enforced
- Identify areas of improvement – *while the process is in use*
- Measure deviations and see the risks, issues, and outcomes

- Extend the use of development tools
  - SCM tools do this, recently, but lack sufficient complexity
  - Few are truly enterprise-grade
  - Vendor lock-in
  
- Substitute a business process management system
  - Too simplistic
  
- Use project management practices and tools
  - Still doing manual enactment and monitoring – huge responsibility for project managers
  - Prime breeding ground for “process resentment”
  - Does not separate methodology from process

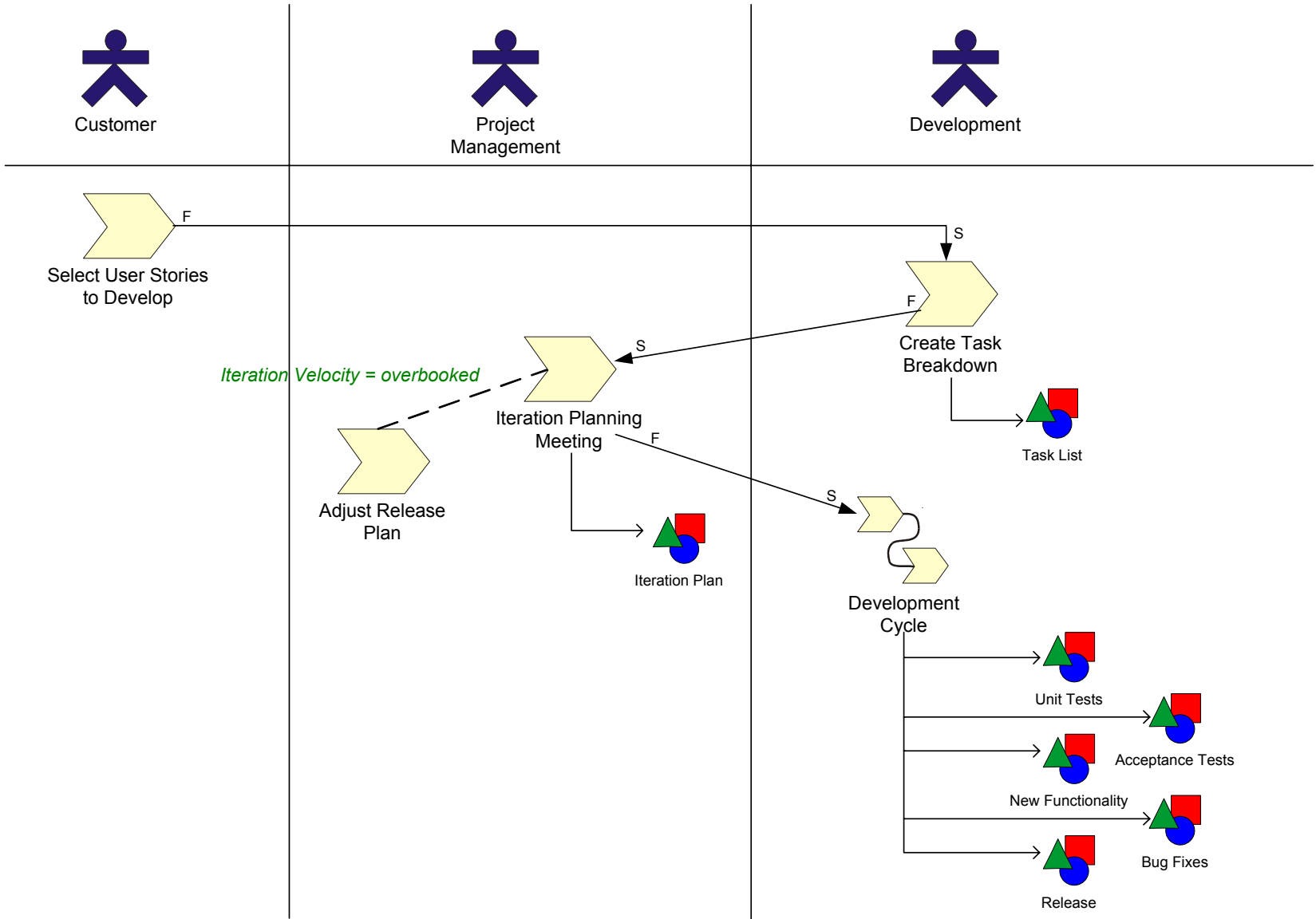
- What is a process?
- Why process models?
- SPEM – the highlights
- Process Automation
- The Future

- SPEM 2.0 RFP is being worked on
  - [spem2rfp@omg.org](mailto:spem2rfp@omg.org)
- Eventual alignment with Business Process Definition Metamodel is a long-term goal.
- Starting to see SPEM-enabled process modeling tools in the marketplace. Process automation is not far away!
- Vendors like IBM/Rational, Softeam are implementing SPEM, and more organizations are starting to use it to model their processes.

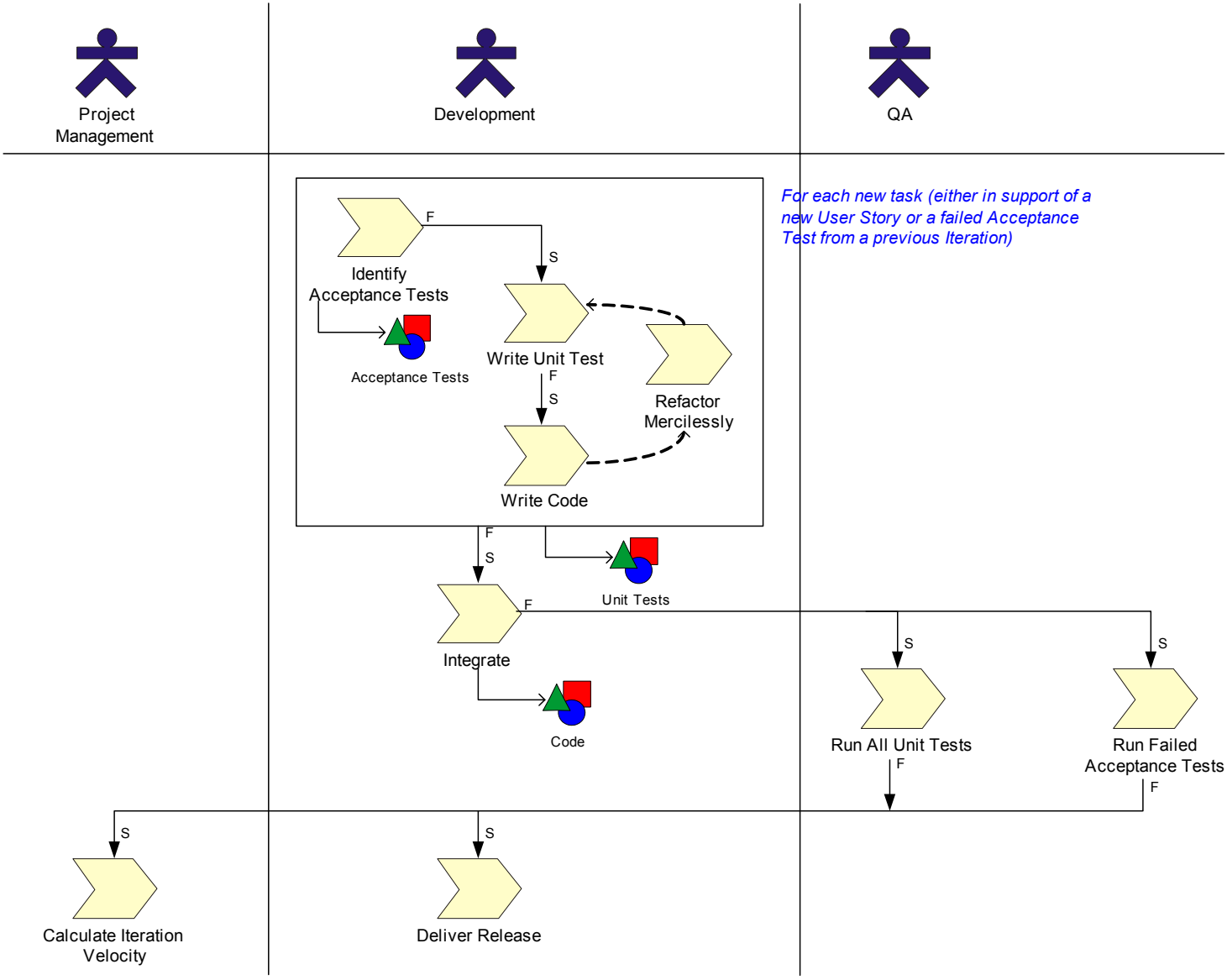
# eXtreme Programming



# Iteration



# Development Cycle



*For each new task (either in support of a new User Story or a failed Acceptance Test from a previous Iteration)*