Patterns in Software Engineering

Lecturer: Raman Ramsin

Lecture 15

Process Patterns

Process Patterns

- Results of applying abstraction to recurring processes and process components
- Create means for developing methodologies through composition of appropriate pattern instances
- Reflect the state of the practice and are based on well-established, refined concepts

Process Patterns: Coplien

- The first recorded reference to the term “Process Pattern” was made by Coplien in his landmark paper in 1994.
- Coplien defined process patterns as “the patterns of activity within an organization (and hence within its project)”.
- Almost all his patterns are relatively fine-grained techniques for exercising better organizational and management practices.
- Do not constitute a comprehensive, coherent whole for defining a software development process.

Process Patterns: Ambler

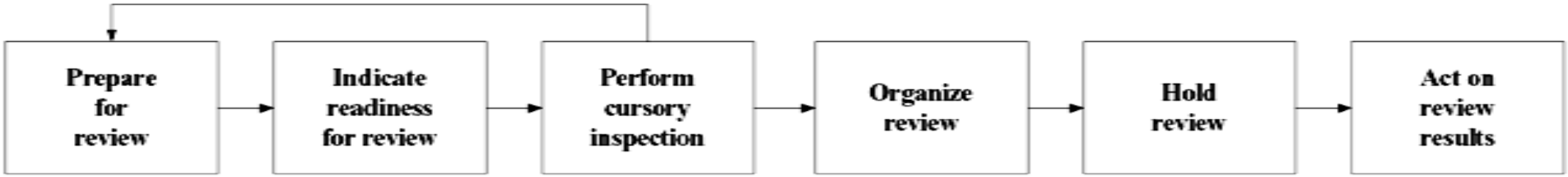
- Ambler is the author of the only books so far written on object-oriented process patterns.
- Defines a process pattern as “a pattern which describes a proven, successful approach and/or series of actions for developing software”
- Defines an object-oriented process pattern as “a collection of general techniques, actions, and/or tasks (activities) for developing object-oriented software”.

Ambler's Process Patterns: Types

- In the ascending order of abstraction level:
  1. *Task Process Pattern:* depicting the detailed steps to execute a specific *task* of the process.
  2. *Stage Process Pattern:* depicting the steps that need to be done in order to perform a *stage* of the process. A *stage* process pattern is usually made up of several *task* process patterns.
  3. *Phase Process Pattern:* depicting the interaction of two or more *stage* process patterns in order to execute the *phase* to which they belong.
- In any process, *phases* are performed in serial order, whereas the *stage* patterns inside them can be executed iteratively.
- Ambler proposes many patterns of each type, complete with detailed steps and guidelines for integrating and shaping the patterns into a comprehensive process.

Ambler's Process Patterns: Task – Example

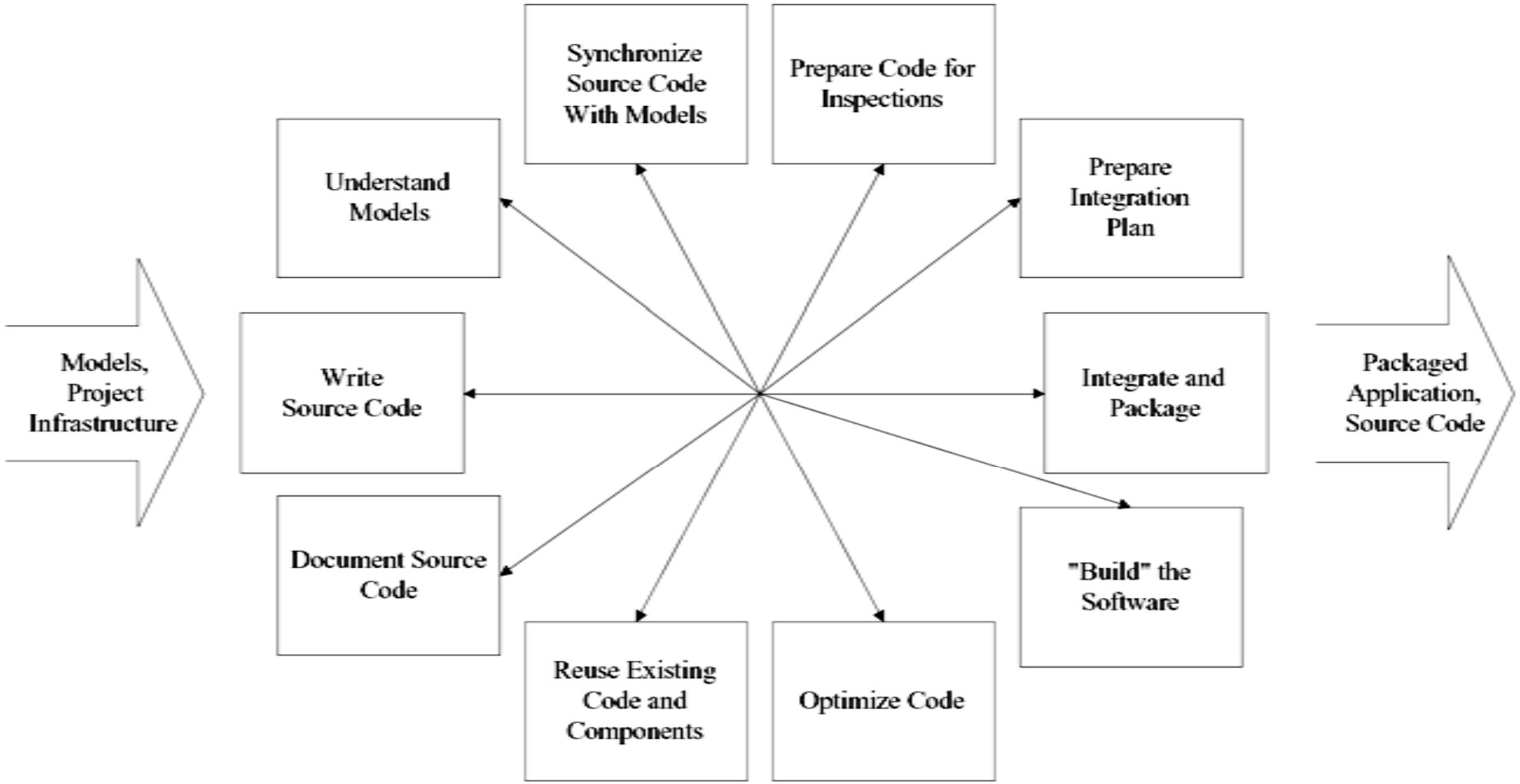
Technical Review



[Ambler 1998]

Ambler's Process Patterns: Stage – Example

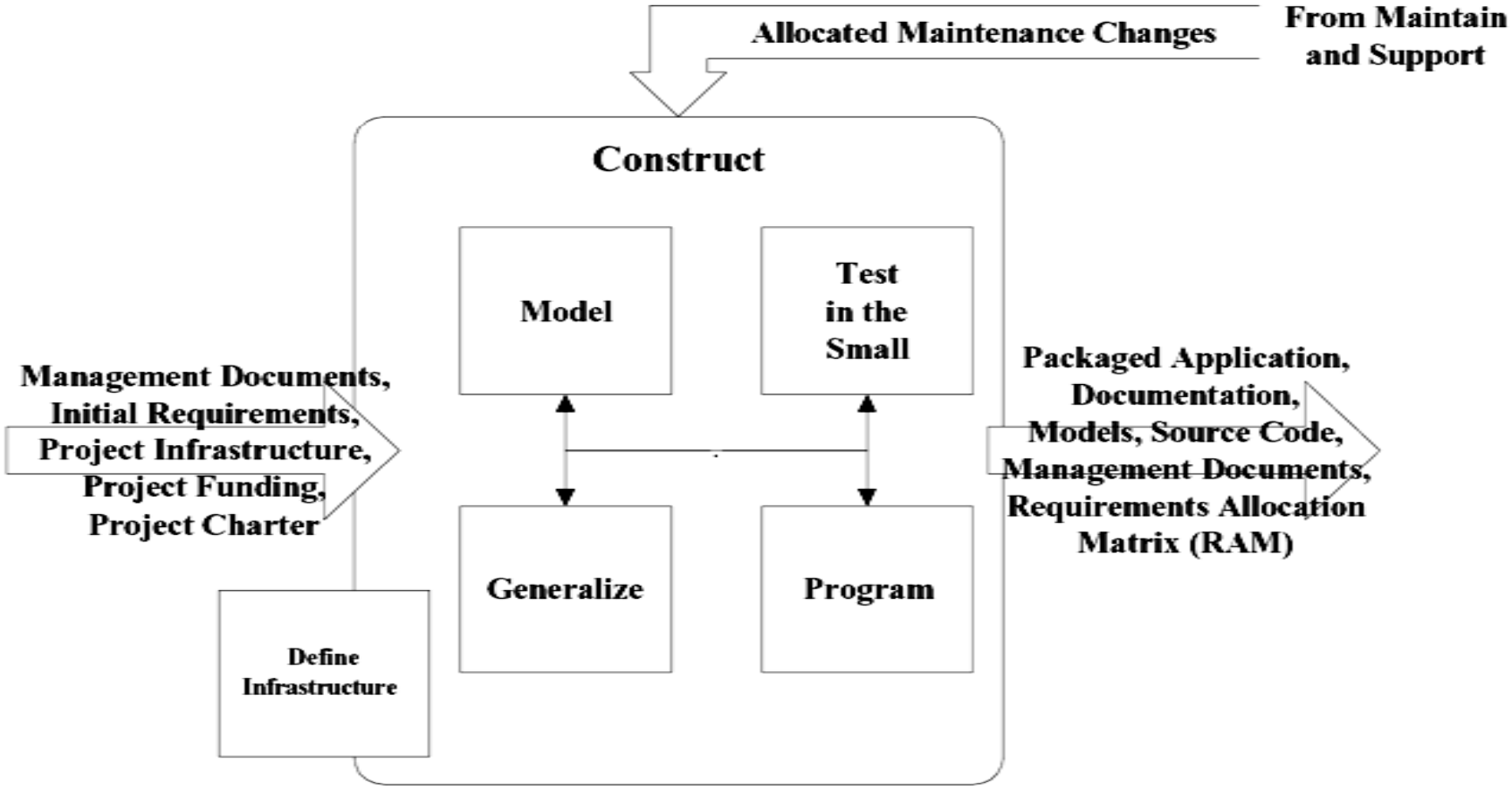
Program



[Ambler 1998]

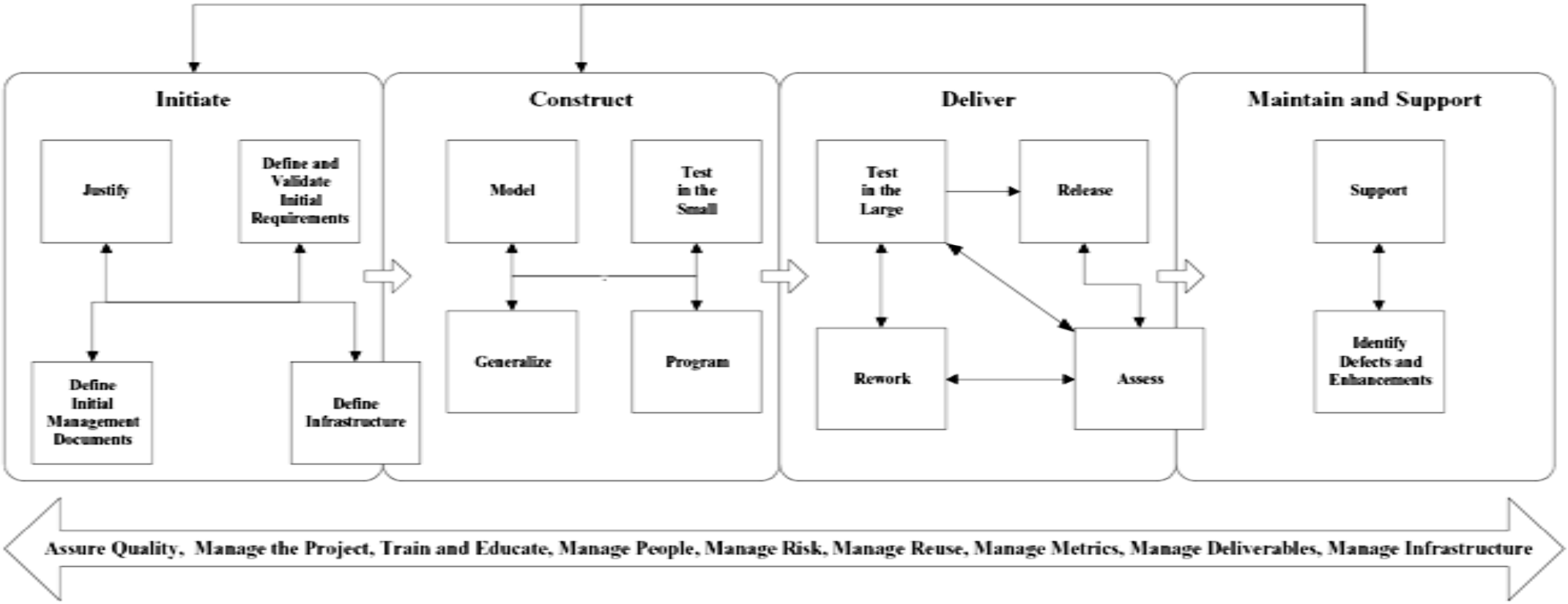
Ambler's Process Patterns: Phase – Example

Construct



[Ambler 1998]

Object Oriented Software Process (OOSP)



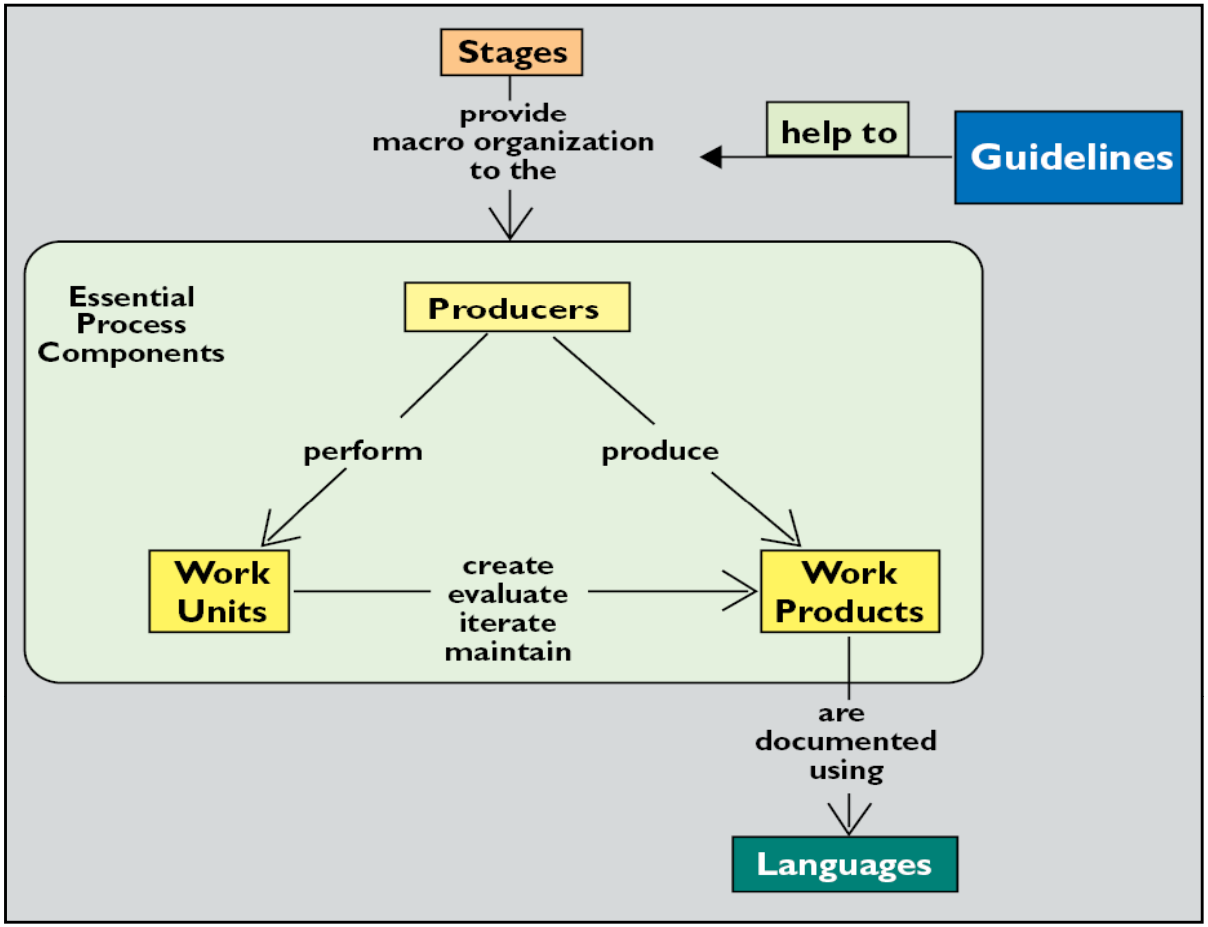
[Ambler 1998]

Object-oriented Process, Environment and Notation (OPEN)

- First introduced in 1996 as the result of the integration of four methodologies: MOSES, SOMA, Synthesis and Firesmith; later deeply influenced by BON and OOram
- Presented as a framework called OPF (OPEN Process Framework)
- Contains a process component library
  - individual process-component instances can be selected and put together to create a specific process instance
- Some process components are in fact process patterns

OPEN Process Framework (OPF)

- A process metamodel defining five classes of components and guidelines for constructing customized OPEN processes



For each element (represented by box), OPEN permits the user to select how many and which instances will be used. The OPF documentation provides a comprehensive list of suggestions on the best selections together with guidelines on their best organization.

[Firesmith and Henderson-Sellers 2001]

OPF: Component Classes

- *Work Products:* any significant thing of value (document, diagram, model, class, application) developed during the project.
- *Languages:* the media used to document work products, such as natural languages, modeling languages such as UML or OML, and implementation languages such as Java, SQL, or CORBA-IDL.
- *Producers:* active entities (human or nonhuman) that develop the work products.
- ***Work Units:*** operations that are performed by producers when developing work products. One or more *producers* develop a *work product* during the execution of one or more *work units*.
- ***Stages:*** durations or points in time that provide a high-level organization to the work units.

OPF: Work Units

- ***Activity:***
  - a major work unit consisting of a related collection of jobs that produce a set of work products
  - Coarse-grained descriptions of what needs to be done
  - Some important instances defined by OPEN are: Project Initiation, Requirements Engineering, Analysis and Model Refinement, Project Planning, and Build
- ***Task:***
  - Smallest atomic unit of work
  - Small-scale jobs associated with and comprising the activities
  - Resulting in the creation, modification, or evaluation of one or more work products
- ***Technique:***
  - Define how the jobs are to be done
  - Ways of doing the tasks and activities

OPF Repository

- Contains a range of predefined instances for each class and subclass in the OPF metamodel; e.g.:
  - 30 predefined instances of Activity
  - 160 instances of Task
  - 200 instances of Techniques
  - 76 instances of Role

OPEN: Process Instantiation

- The following tasks are performed (through applying the guidelines proposed by OPF) in order to instantiate, tailor and extend an OPEN process:

1. *Instantiating* the OPEN library of predefined component-classes to produce actual process components
2. *Choosing* the most suitable process components from the set of instantiated components
3. *Adjusting* the fine detail inside the chosen process components
4. *Extending* the existing class library of predefined process components to enhance reusability

OPF: Task-Activity Matrix

Task	Activity					
	1	2	3	4	5	6
Code		x				
Construct the object model		x				x
Develop and implement resource allocation plan						
develop iteration plan	x					
develop timebox plan	x					
set up metrics collection program	x					
specify quality goals	x					
Evaluate quality			x	x		x
Identify CIRTs (Class, Instance, Role, or Type)		x				
Map roles onto classes		x(OOP)				
Test			x	x	x	
Write manuals and other documentation			x	x	x	x

key

1. Project planning
2. Modeling and implementation: OO analysis, design, programming
3. Verification and validation
4. User review
5. Consolidation
6. Evaluation

[Henderson-Sellers 2003]

OPF: Technique-Task Matrix

Techniques	Task 1	2	3	4	5	6	7
Abstract class identification		x					
Abstraction utilization		x			x		
Class internal design	x	x					
Class naming		x		x			
Collaborations analysis		x					
Complexity measurement				x			
Contract specification	x	x			x		
Coupling measurement				x			
CRC card modeling		x			x		
Generalization and inheritance identification		x					x
Implementation of services	x						
Implementation of structure	x					x	
Inspections				x			x
Interaction modeling		x					
Package and subsystem testing							x
Prototyping	x	x					
Relationship modeling	x	x					
Responsibility identification	x			x	x		x
Role modeling		x				x	
Service identification		x					
State modeling		x					
Textual analysis					x		
Timeboxing			x				
Unit testing							x
Walkthroughs				x			x

key

1. Code
2. Construct the object model
3. Develop and implement resource allocation plan
4. Evaluate quality
5. Identify CIRTs
6. Map roles onto classes
7. Test

[Henderson-Sellers 2003]

References

- Coplien, J. O., A development process generative pattern language. In *Proceedings of the First Annual Conference on Pattern Languages of Programming (PLoP)*, 1994.

- Ambler, S. W., *Process Patterns: Building Large-Scale Systems Using Object Technology*. Cambridge University Press, 1998.

- Firesmith, D., Henderson-Sellers, B., *The OPEN Process Framework: An Introduction*. Addison-Wesley, 2001.

- Henderson-Sellers, B., Method engineering for OO systems development. *CACM 46,* 10 (October), 73-78, 2003.