IN5140: Process Improvement and Agile Methods in Systems Development

Lecture 29 August 2024:

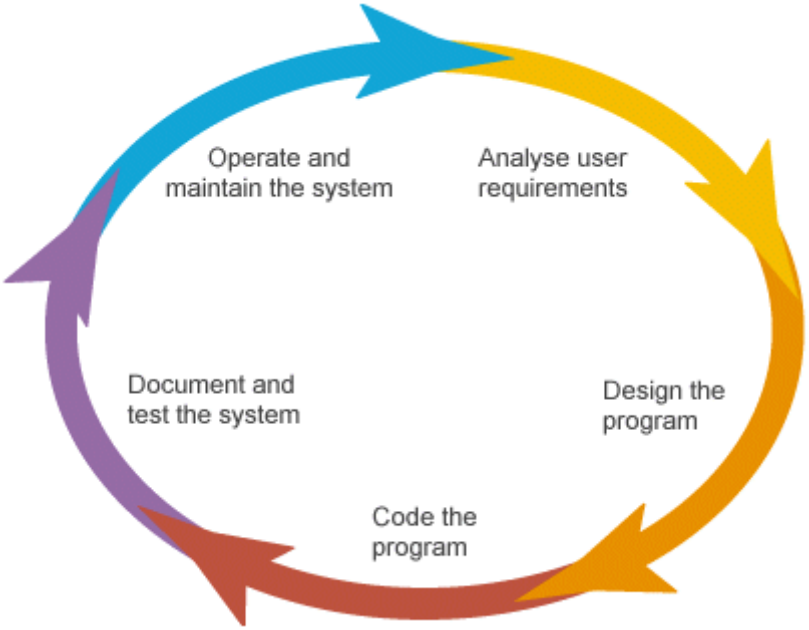
Process Modeling

Yngve Lindsjørn

E-mail: ynglin@ifi.uio.no

Software Development

- Overall topic of IN5140:
  How to develop software systems?
  - Software development includes all activities involved between the conception of the desired software and the operationalization of it
  - May include new development, prototyping, modification, reuse, maintenance, or any other activities that result in software products



UNIVERSITETET I OSLO

Software Maintenance = All Changes Made to a System after it has Become Operational (Put into Use)

- Types of software maintenance
  - Fixing bugs (corrective maintenance)
  - Improving structure (re-engineering, refactoring) (preventive maintenance)
  - Adapt to changing environments (adaptive maintenance)
  - New functionality (perfective maintenance)

- Constitutes most of the costs in the total lifetime of a system
- Sometimes implicit in the definition of software development, sometimes not

UNIVERSITETET I OSLO

Software is widespread and used in

- industry
  - transportation, energy, telecom, medical, fishing, agriculture, etc.
- administration
  - banking, insurance, e-government, information systems, payroll systems, HR (human resource) systems, ERP (enterprise resource planning) systems, …
- science
  - medicine, biology, chemistry, physics, …
- consumers
  - entertainment, education, information, assistance, …
- …

Variation in Size and Complexity

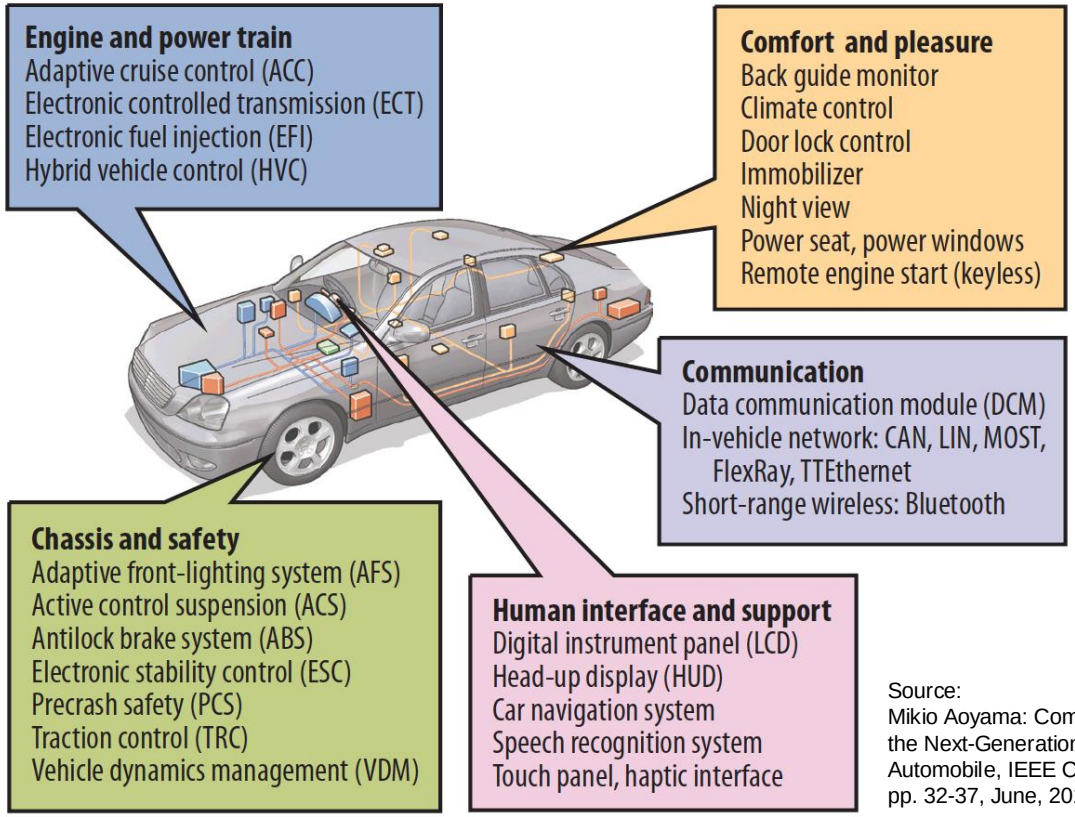
- Software
  - From a few thousands to millions lines of code (LOC)
- Development teams
  - From 1 to more than 1000 developers (MS Windows 7)
- Costs (development and maintenance)
  - From a few thousands to billions

Operating systems:
Linux kernel: 15 M LOC
Windows 10: 50 M LOC
MacOS 10.4: 86 M LOC

UNIVERSITETET I OSLO

Example of a Software-Intensive System

- Example: Embedded systems in a car



Source:
Mikio Aoyama: Computing for the Next-Generation Automobile, IEEE Computer, pp. 32-37, June, 2012

What's Special about Software?

- Software is developed, not produced – every software product is unique
- Software development
  - is a design process,
  - is often experimental, and
  - requires knowledge, skills and creativity
- Software is not tangible (not "physical")
  - Natural laws irrelevant (only relevant for HW)
  - Often large design teams
  - High degree of flexibility/changeability of artefacts
  - Measurements of size, complexity and quality is not (yet?) well-defined (different to, for example, physics)

"Soft Factors" dominate!

UNIVERSITETET I OSLO

Challenges with Software/Systems

- Unsatisfactory quality (incl. functionality)
- Exceeding budget (cost/effort)
- Late delivery (lead time problems)

Source:
”Extreme Chaos" by The Standish Group (2001):

Reasons for Software Project Failures

Table 3

Reasons for project cancellation with percentages and 95% confidence intervals for the 2007 respondents (n = 18)*

Reason for cancellation	Percentage of respondents (95% confidence interval)
Senior management not sufficiently involved	33 (13, 59)
Too many requirements and scope changes	33 (13, 59)
Lack of necessary management skills	28 (10, 54)
Over budget	28 (10, 54)
Lack of necessary technical skills	22 (6, 48)
No more need for the system to be developed	22 (6, 48)
Over schedule	17 (4, 41)
Technology too new; didn't work as expected	17 (4, 41)
Insufficient staff	11 (1, 35)
Critical quality problems with software	11 (1, 35)
End users not sufficiently involved	6 (0, 27)

*The 95% confidence interval is usually wide because we're looking at only 18 cancelled projects. The respondents had the option of adding qualitative information as well as the predefined categories.

K. El-Emam and A.G. Kouru. A Replicated Survey of IT Software Project Failures, IEEE Software, 25(5):84-90, 2008.

SW Project Risks
(from 12 surveys)

Source:
Top Ten Lists of Software Project Risks: Evidence from the Literature Survey by Tharwon Arnuphaptrairong, Proceedings of IMEC 2011, March 16-18, 2011, Hong Kong

TABLE VI
TOP TEN LIST OF SCHMIDT ET AL. [6], (USA)

Rank	Software Risk	Dimension
1	Lack of top management commitment to the project	Org
2	Misunderstanding of the requirements	R
3	Not managing change properly	P&C
4	Failure to gain user commitment	U
5	Lack of effective project management skill	P&C
6	Lack of adequate user involvement	U
7	Failure to manage end-user expectations	U
	Lack of effective project management methodology	P&C
	Unclear / misunderstood scope/ objectives	P&C
	Changing scope / objectives	P&C

TABLE VI
TOP TEN LIST OF SCHMIDT ET AL. [6], (FINLAND)

Rank	Software Risk	Dimension
1	Lack of effective project management skill	P&C
2	Lack of top management commitment to the project	Org
3	Lack of required knowledge /skill in the project personnel	T
4	Not managing change properly	P&C
5	No planning or inadequate planning	P&C
6	Misunderstanding of the requirements	R
7	Artificial deadlines	P&C
8	Failure to gain user commitment	U
9	Lack of frozen requirements	R
10	Lack of "people skill" in project leadership	T

TABLE XI
SOFTWARE RISKS FREQUENCY BY DIMENSION

Dimension	Total frequency
User	14
Requirement	17
Complexity	4
Planning and Control	**27**
Team	9
Organizational Environment	9
Total	80

UNIVERSITETET I OSLO

Goal of IN5140

Teach you to help avoid software project failures by

- identifying risks and
- means to tackle them

What is a Process?

- – a set of activities carried out to fulfil a job
- Usually wish to run a process that is carefully planned
- Everything we do involves processes, for example:

  - **Plan a project**
  - **Prepare a meeting**
  - **Run a meeting**
  - **Write a requirements specification**
  - **Design a web-site**
  - **Program a function**
  - **Run a test**

UNIVERSITETET I OSLO

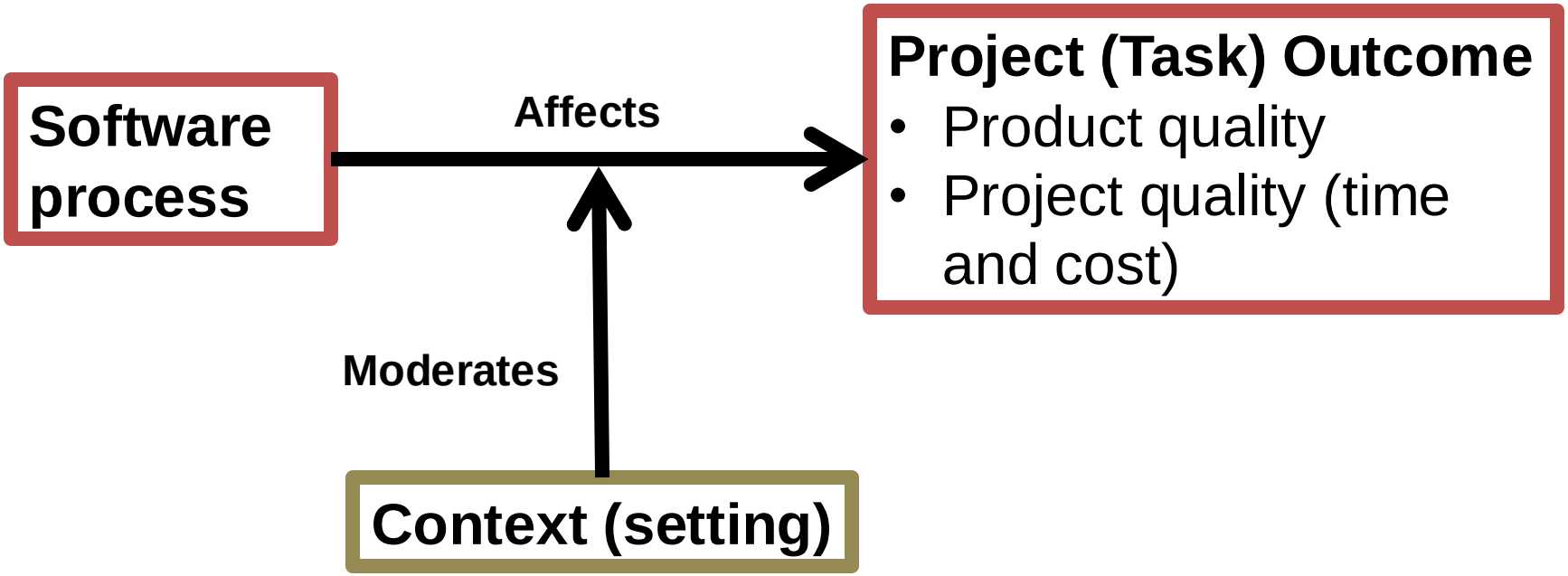
System Development Process

- System development process (= software process): the activities that are performed to develop a software system and the ***way they are performed***
- The activities vary but will always include the elements:
  - Specification of requirements, that is, what the system should do and the quality of how it is done
    - the way requirements are specified may vary, which leads to different requirement specification processes
  - Design of the system
  - Coding (programming)
  - Validation that the system satisfies the needs of the customer or user
  - Evolution of the system, that is, changes according to new or changed requirements

Process Affects Properties of both Project and System being Developed

- The development process and the context in which it is performed affect the quality of project properties and the resulting system or product being developed
- The process also affects the work environment (work satisfaction, motivation, competence development, etc.), which in turn affects project and product quality

Process and Project (Task) Outcome



UNIVERSITETET I OSLO

Aspects of Process

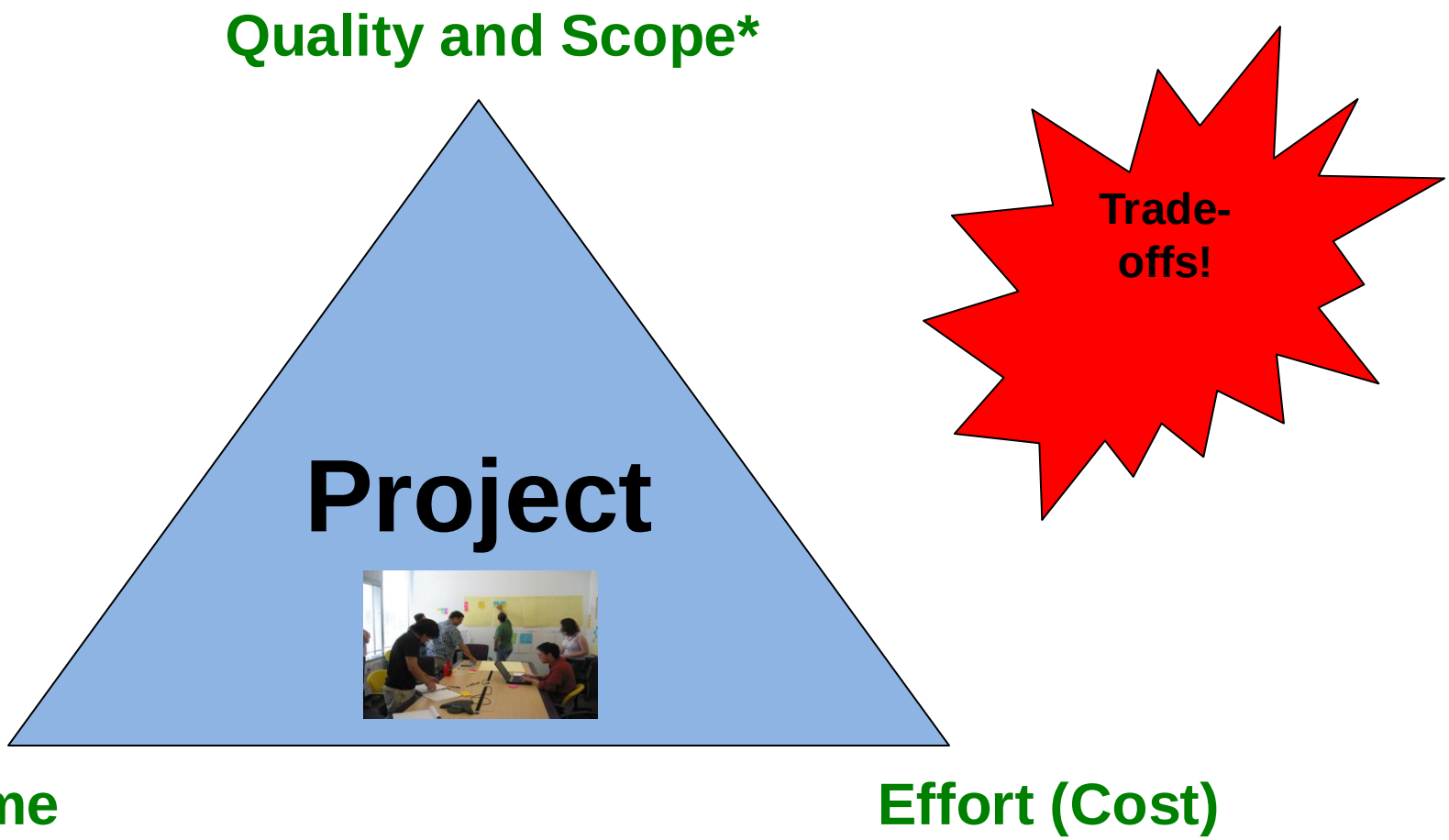
- Which activities are in focus in the process?
- How much effort is spent on each of them?
- A process may also include
  - Which and the way methods, techniques, tools and practices are used
  - Parts of the products/results of an activity
  - Roles of those involved in the process

Examples of Roles

- Developer
- Maintainer
- Architect/System designer
- Graphical designer
- Documenter
- Tester
- Project manager
- User/customer representative

Employing a project with the right competence is far from trivial!

The Magic Triangle



*the extent of the functionality of the system

UNIVERSITETET I OSLO

Agile Roles (from the text book chapter 5)

- Manager
  - A supporting role (not assigning tasks). “Establishing an environment that enables the team to work successfully”
- Product owner
  - Facilitates decisions about the product. Select user stories from the product backlog. Evaluates the result of the sprint (in Scrum)
- Team
  - Group of people, but also viewed as a “single” character. Self-organized. Cross-functional (teams should be formed along the lines of features)
- Members and Observers
  - The observers will give their opinion if invited, but project decisions, such as including functionality, are the privilege of members
- Customer
  - “Put the customer at the centre”. In Scrum often the Product Owner represents the customer
- Coach, Scrum Master
  - The Scrum master is responsible for making sure a Scrum team lives by the values and practises of Scrum

UNIVERSITETET I OSLO

Examples of Tools

Tools for:

- Development (IDE) (e.g. Android Studio)
- Configuration/change management (e.g. GIT)
- Testing
- Diagram construction (e.g. UML, BPMN)
- Project management
- Bug & issue tracking (e.g. JIRA, GITHUB)
- Collaboration

Choice of tools is not trivial

UNIVERSITETET I OSLO

Some Collaboration tools used in IN2000 (40 teams)

- Github
- Google hangouts
- Zoom
- Teams
- Slack
- Discord
- Facebook/Messenger
- Mozilla hubs
- Google hangouts
- Monday
- Trello
- Notion
- Google Docs, Google drive
- Invition
- Figma
- …….

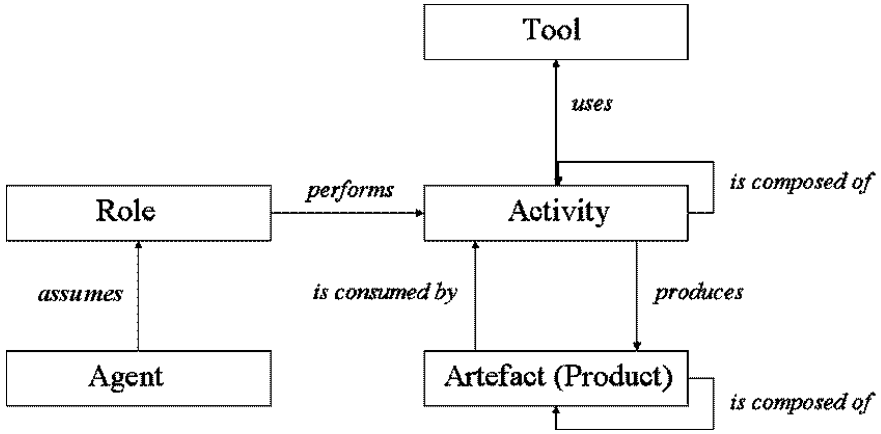
Project process model

- Ideally, a process should be planned, tailored to the context and adjusted based on experience. One then needs a process model
- A process model defines **Who** is doing **What**, **When** and **How** to reach a specific goal.
  - In software engineering the goal is to build a software product or to enhance an existing one
- A good process model…
  - provides guidelines for efficient development of quality software
  - reduces risk and increases predictability
  - promotes common vision and culture

Attributes of a project process model

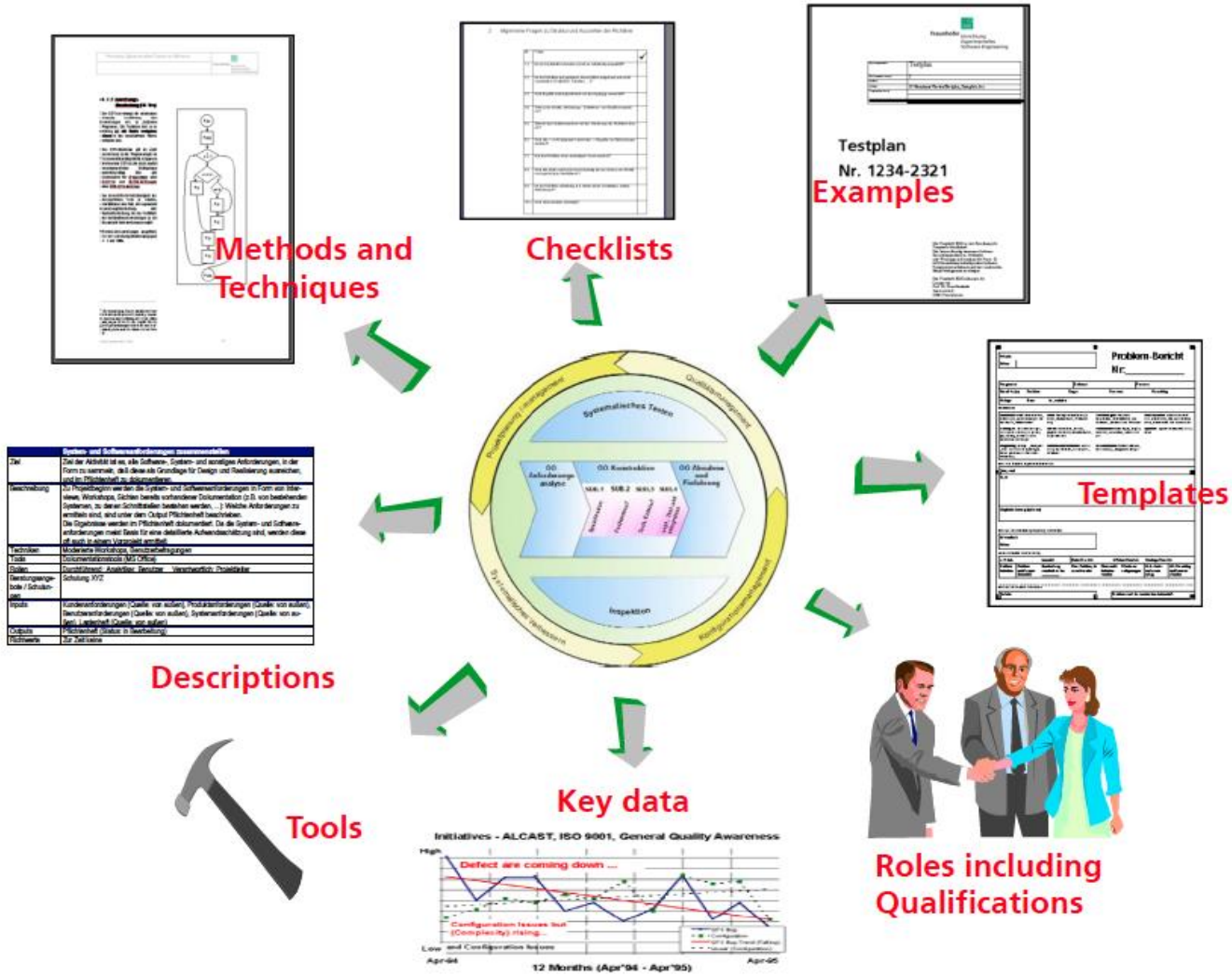
- The activities of the process
  - The order and scope of the activities
  - The pre- and post conditions of activities
  - Whether the activities are performed one or several times
- The roles involved in the process
- The artifacts/deliverables of the process (models, documents, source code, etc.)
- The methods, practices, tools and techniques involved in the process

Model



UNIVERSITETET I OSLO

Typical elements of a project process model



UNIVERSITETET I OSLO

Purpose of (Descriptive) Process Modeling

- Understand the process
- Communicate (about) the process
- Support management
- Guide the work
- Identify typical deviations from the prescribed process models
- Improve software development activities
- Support measurement

Modeling Languages and tools

- Business Process Modeling Notation (BPMN) is the most common process modeling language (www.bpmn.org/); that **is, a language for creating process models.**
- **In IN5140, we use BPMN for creating *software process models***
- Some other modelling languages for process modelling are:
- Flowcharts that can be drawn in many different tools
- IDEF which is a family of modeling languages (http://www.idef.com/)
- Activity diagrams of the Unified Modeling Language (UML)
  - **Used in INF1050 and IN1030 (Software Engineering)**

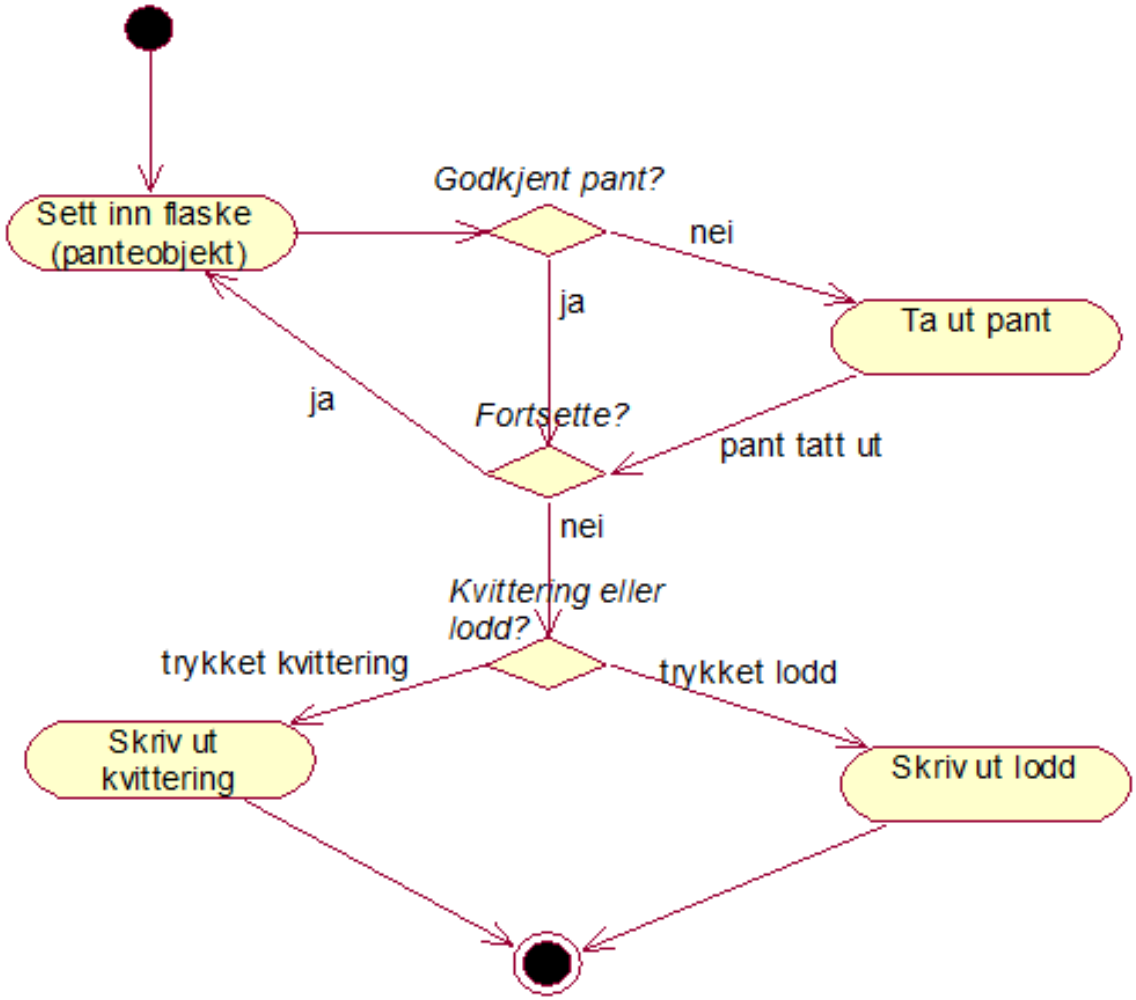
UNIVERSITETET I OSLO

BPMN – Business Process Model and Notation

- First published in 2002 – version today is BPMN 2.0
- Official standard for Object Management Group (OMG) from 2006
- SAP, Oracle and IBM have worked with specification since 2007
- Links:
  - http://www.bpmn.org
  - http://bpmb.de/poster
- Many tools implement BPMN
  - Draw.io (easy to use)
  - Ludichart (www.ludichart.com)
  - Yaoqiang BPMN Editor is used in some of the examples in this lecture. Open Source and free for use.
    - http://sourceforge.net/projects/bpmn
  - Signavio is also used (and is much used in practise by companies etc.). Free 30 days trial. (http://signavio.com)

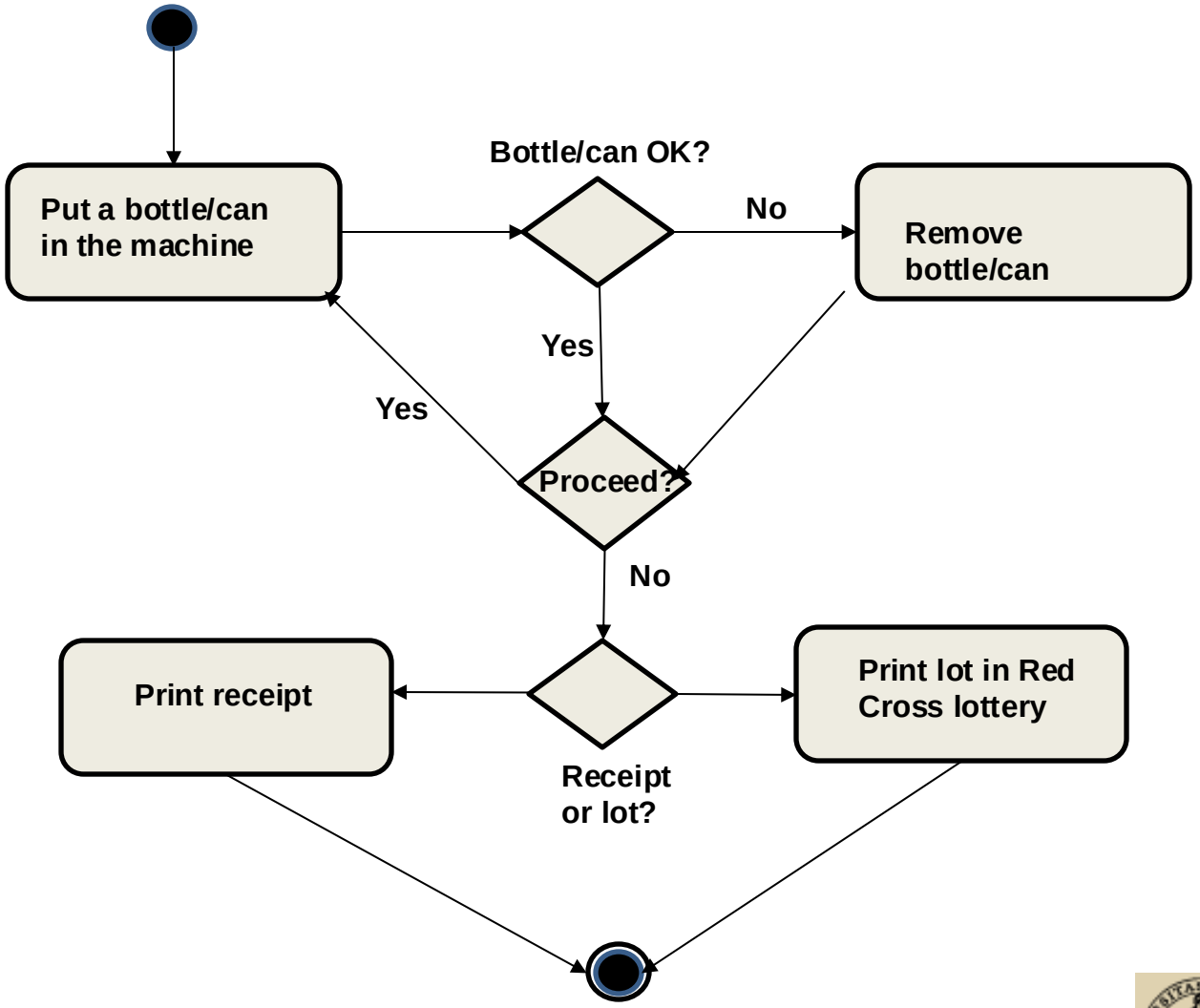
UNIVERSITETET I OSLO

From INF1050/IN1030: Activity Diagram (UML) – Deliver bottles/cans in a machine (“pante flasker»)



UNIVERSITETET I OSLO

From INF1050/IN1030: Activity Diagram (UML) – Deliver bottles/cans in a machine (“pante flasker»)



UNIVERSITETET I OSLO

BPMN 2.0 - Business Process Model and Notation

http://bpmb.de/poster

Activities

A **Task** is a unit of work, the job to be performed. When marked with a [+] symbol it indicates a **Sub-Process**, an activity that can be refined.

A **Transaction** is a set of activities that logically belong together; it might follow a specified transaction protocol.

An **Event Sub-Process** is placed into a Process or Sub-Process. It is activated when its start event gets triggered and can interrupt the higher level process context or run in parallel (non-interrupting) depending on the start event.

A **Call Activity** is a wrapper for a globally defined Task or Process reused in the current Process. A call to a Process is marked with a [+] symbol.

Activity Markers

Markers indicate execution behavior of activities:

- Sub-Process Marker
- Loop Marker
- Parallel MI Marker
- Sequential MI Marker
- Ad Hoc Marker
- Compensation Marker

Task Types

Types specify the nature of the action to be performed:

- Send Task
- Receive Task
- User Task
- Manual Task
- Business Rule Task
- Service Task
- Script Task

Sequence Flow
defines the execution order of activities.

Default Flow
is the default branch to be chosen if all other conditions evaluate to false.

Conditional Flow
has a condition assigned that defines whether or not the flow is used.

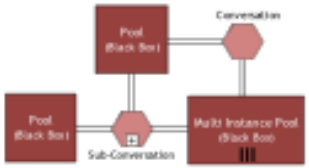
Conversations

A **Conversation** defines a set of logically related message exchanges. When marked with a [+] symbol it indicates a **Sub-Conversation**, a compound conversation element.

A **Call Conversation** is a wrapper for a globally defined Conversation or Sub-Conversation. A call to a Sub-conversation is marked with a [+] symbol.

A **Conversation Link** connects Conversations and Participants.

Conversation Diagram



Choreographies

A **Choreography Task** represents an interaction (Message Exchange) between two Participants.

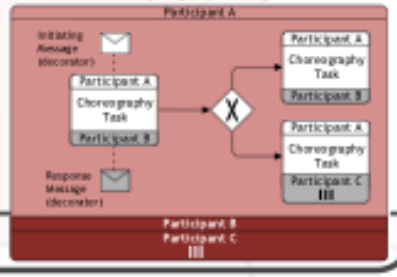
A **Sub-Choreography** contains a refined choreography with several interactions.

A **Call Choreography** is a wrapper for a globally defined Choreography Task or Sub-Choreography. A call to a Sub-Choreography is marked with a [+] symbol.

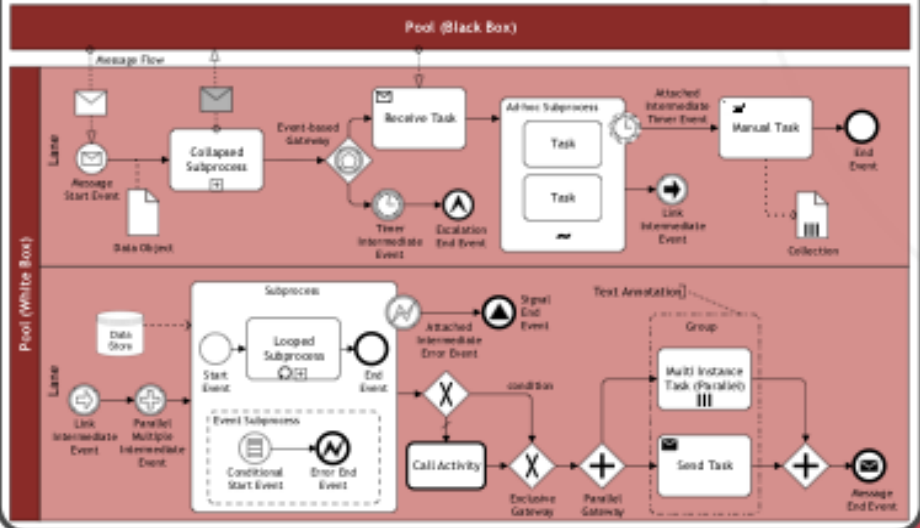
Multiple Participants Marker denotes a set of Participants of the same kind.

Message a decorator depicting the content of the message. It can only be attached to Choreography Tasks.

Choreography Diagram



Collaboration Diagram



Events

	Start: Standard	Start: Event Sub-Process Interrupting	Start: Event Sub-Process Non-Interrupting	Intermediate: Catching	Intermediate: Boundary Interrupting	Intermediate: Boundary Non-Interrupting	Intermediate: Throwing	End: Standard
None: Untyped events, indicate start point, state changes or final states.	○						○	○
Message: Receiving and sending messages.	✉	✉	✉	✉	✉	✉	✉	✉
Timer: Cyclic timer events, points in time, time spans or timeouts.	🕒	🕒	🕒	🕒	🕒	🕒		
Escalation: Escalating to an higher level of responsibility.		Ⓐ	Ⓐ		Ⓐ	Ⓐ	Ⓐ	Ⓐ
Conditional: Reacting to changed business conditions or integrating business rules.	☰	☰	☰	☰	☰	☰		
Link: Off-page connectors. Two corresponding link events equal a sequence flow.				⇨			➡	
Error: Catching or throwing named errors.		⚡			⚡			⚡
Cancel: Reacting to cancelled transactions or triggering cancellation.					⊗			⊗
Compensation: Handling or triggering compensation.		◁◁			◁◁		◀◀	◀◀
Signal: Signalling across different processes. A signal thrown can be caught multiple times.	△	△	△	△	△	△	▲	▲
Multiple: Catching one out of a set of events. Throwing all events defined	⬠	⬠	⬠	⬠	⬠	⬠	⬟	⬟
Parallel Multiple: Catching all out of a set of parallel events.	✚	✚	✚	✚	✚	✚		
Terminate: Triggering the immediate termination of a process.								●

Gateways

Exclusive Gateway
When splitting, it routes the sequence flow to exactly one of the outgoing branches. When merging, it awaits one incoming branch to complete before triggering the outgoing flow.

Event-based Gateway
Is always followed by catching events or receive tasks. Sequence flow is routed to the subsequent event/task which happens first.

Parallel Gateway
When used to split the sequence flow, all outgoing branches are activated simultaneously. When merging parallel branches it waits for all incoming branches to complete before triggering the outgoing flow.

Inclusive Gateway
When splitting, one or more branches are activated. All active incoming branches must complete before merging.

Exclusive Event-based Gateway (Instantiate)
Each occurrence of a subsequent event starts a new process instance.

Complex Gateway
Complex merging and branching behavior that is not captured by other gateways.

Parallel Event-based Gateway (Instantiate)
The occurrence of all subsequent events starts a new process instance.

Swimlanes

Pools (Participants) and **Lanes** represent responsibilities for activities in a process. A pool or a lane can be an organization, a role, or a system. Lanes subdivide pools or other lanes hierarchically.

Message Flow symbolizes information flow across organizational boundaries. Message flow can be attached to pools, activities, or message events. The Message Flow can be decorated with an envelope depicting the content of the message.

The order of message exchanges can be specified by combining message flow and sequence flow.

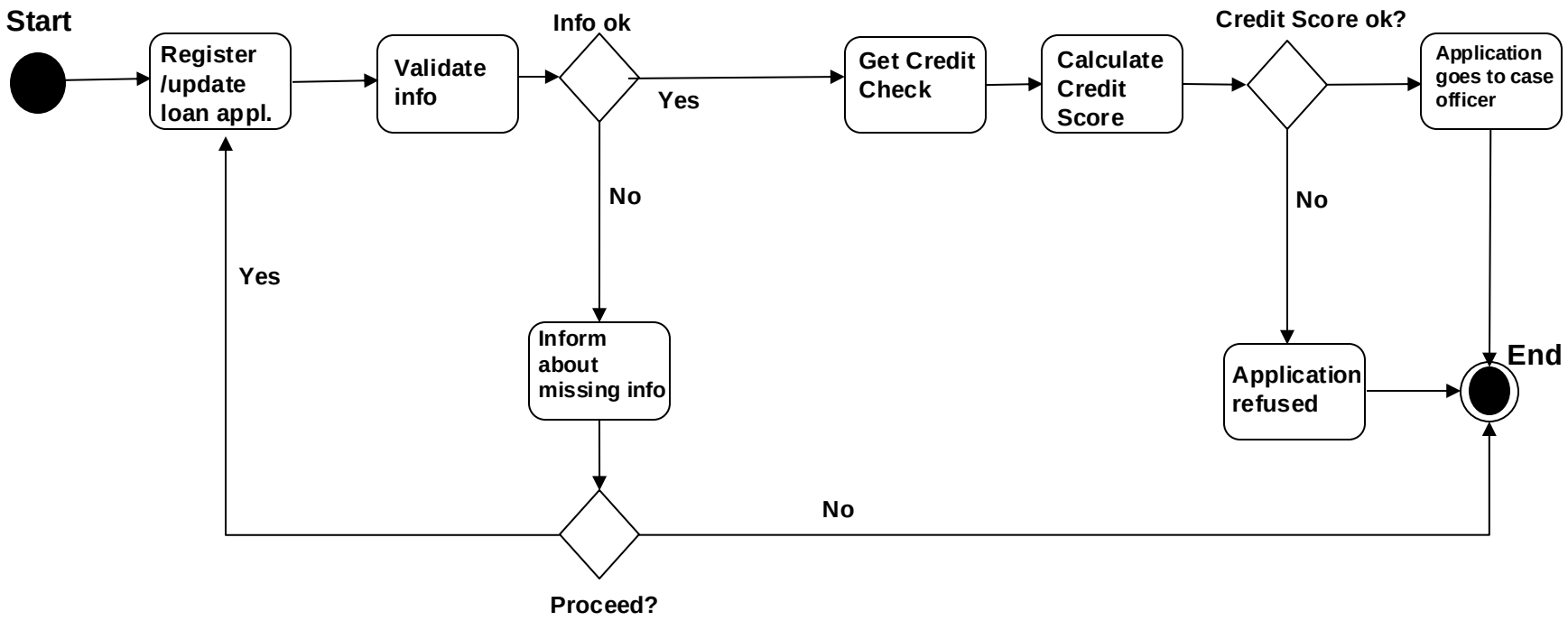
Data

- **A Data Object** represents information flowing through the process, such as business documents, e-mails, or letters.
- **A Collection Data Object** represents a collection of information, e.g., a list of order items.
- **A Data Input** is an external input for the entire process. A kind of input parameter.
- **A Data Output** is data result of the entire process. A kind of output parameter.
- **A Data Association** is used to associate data elements to Activities, Processes and Global Tasks.
- **A Data Store** is a place where the process can read or write data, e.g., a database or a filing cabinet. It persists beyond the lifetime of the process instance.

BPM Offensive Berlin · HPI Hasso Plattner Institut · camunda · inubit · SIGNAVIO

© 2011

From INF1050/IN1030: Activity diagram "Registrer lånesøknad" (register loan application)



UNIVERSITETET I OSLO

Exercise

- Discuss and make a BPMN diagram of the «Register loan» usecase. Use the actors Loaner (User), Credit company and case officer (lånekonsulent). Extend the task so that documents are sent for signing.

Example of process modelling – eating dinner

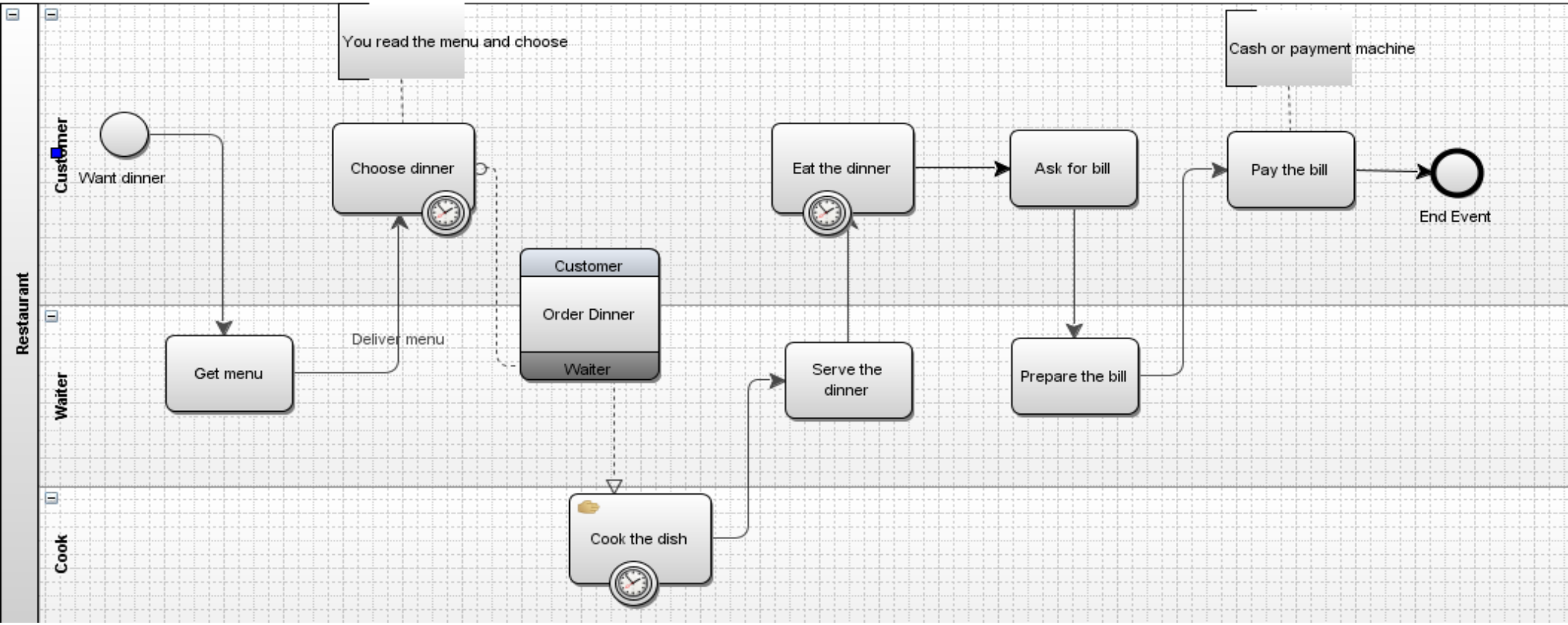
Suppose you are in a restaurant. You have found a table and want to have dinner:

> You get a menu from a waiter, choose a dish and make an order. A waiter writes it down, goes to the kitchen and requests the cook to make your dish. When the dish is ready, the cook informs waiter and waiter serves the dish to you. You eat the meal and pay for it.

Exercise

- Discuss and make a BPMN diagram of the «Eating dinner» usecase. Call the pool Restaurant and use the actors Customer, Waiter and Cook.

Example – Eating dinner



UNIVERSITETET I OSLO

Exercise in group session

- Read the textual description of the process of developing a new functionality for a web application.
- Use BPMN and create a process model of the process of developing a new functionality for the existing web application.

Exercise – INF5140: Process Modeling using BPMN

Background

- Company X uses a web application to support its business. The web application was developed and is currently maintained by Company Y. The two companies work together on improvements and further development; X identifies needs for new functionality and orders this from Y, and Y regularly delivers new versions of the web application with the requested functionality to X.

The process of developing new functionality

- Company X has a Product owner for the web application who is responsible for identifying new business opportunities, for describing new requirements and for testing. The Product owner regularly hands over a requirement specification to the Delivery manager at Company Y. The Delivery manager is responsible for the successful delivery of new versions of the application. The Delivery manager cooperates with a team of developers in company Y. They discuss the requirements from company X and estimate the tasks. Then the Delivery manager informs the Product owner about the estimates and costs. Based on this information, the Product owner decides which requirements should be prioritized for the new version of the system and communicates this back to the Delivery manager. Then the developers start development. When the new features have been developed, they are deployed to the test environment by the Operation department at Company Y. The Delivery manager requests the Product owner to test that implemented features correspond to the specified requirements. If some of the features do not pass the test, the Product owner makes a list of errors and informs the Delivery manager, who in turn informs the developers of the necessary changes and error corrections. When all tests are passed, the test cases are documented and sent to the Delivery manager, who requests the Operation department to deploy the new version to the production environment. The Operation department deploys the new version.