Appendix 1 – Preliminary Specification, System Functions and Features from the User Perspective

Content

Preliminary remarks:

This document consists of several parts.

Part I depicts the intended system from a more technical perspective, oriented towards the implementation/software engineering work. Already throughout this part, we have added user stories, providing background and insights, described from the perspective of the organizational developers and users of the system. These user stories should describe expected features, functionalities and system behavior. Care has to be taken as the consistency in wordings can't be assured in the user stories as they are authored from a different perspective and by different people.

Parts II to IV are less implementation oriented but rather focus on the backgrounds of the concepts to process management. Parts II and III are purely built up as user stories. In particular, the user story "Functional description from a user perspective" should further depict Einhell's expectations from an organizational point of view.

Note:

Parts II to IV were developed at a much earlier stage. Where descriptions do contradict, Part I shall prevail to the extent of the inconsistency.

PART I: Preliminary User Specifications

Starting Position

"The goal is to develop a process management software that assembles processes out of individual PBBs (process building blocks) in a dynamic manner."

The **process management software** is intended to guide the enactment of agile industrial office processes, involving multiple people (employees, partners, etc.) in multiple teams and locations. The process management software guides the enactment, in that it indicates work packages to be carried out to relevant resources (employees, external participants, etc.) in well-suited **to-do-lists** and supports the execution of the work packages by guiding the work, that has to be carried out by definable SOPs, micro workflows or forms, presenting relevant information for the execution of the work packages and collecting relevant results alongside the execution of the work packages.

Process management systems are neither a workflow system nor a project management tool! Business processes, consisting out of work packages are to be assembled dynamically during enactment time. The work packages are to be modelled in a way that they can function as building blocks, so-called PBBs (process building blocks) that allow a chaining and linking to larger business processes. The process management system supports this chaining and linking and monitors the execution of each work package. It ensures that “ready to process” work packages (PBBs) are indicated at time in the resources’ to-do-lists. The process management system thus ensures sequencing and timing of the work packages to be enacted and thus organizes the build-up of project related knowledge.

In this context, it is assumed that any incoming stimulus to a business operation leads to the response of the organization by developing a sequence of activities (the process) to compensate or utilize the situation that has triggered the stimulus. If the organization derives a response process intent-driven and involves an adequate amount of resources to derive an intended result, we call this a *project*.

Project management systems support resources (project leaders, employees, etc.) by decompositioning problem and target domains and structuring realization approaches. They help with allocation of responsibilities (tasking) and monitor the fulfillment of tasks in very agile processes that need to be coordinated by the resources themselves. The coordination is part of the value-added to be generated with the project. The coordination effort, an inadequately aligned sequence of enactment, a lack of information during execution, the need to manually push and trigger colleagues and co-workers or external parties to contribute at right times etc., are typical for project management

scenarios and thus companies, building their business models on project-organization approaches tend to lose cost-performance-leaderships in competitive business environments.

Highly efficient and thus highly competitive value creation today is known to happen in pre-programed process scenarios. This is typically implemented in industrial production environments, where large amounts of products with identical properties are mass-produced in rigid production environments, enacting pre-defined sequences of work packages (production processes). In these environments the term “project” has disappeared! The project here still would be the creation of a single product but as this creation is implemented to be identical for all products, the term “project” has lost its original sense. In office environments, this organization form was named **workflow organization**. Workflow organizations are implemented based on **workflow management systems** that allow entire processes to be modelled and implemented. Theoretically, this approach delivers best efficiency and performance values on office processes. In practical implementations, indeed, in a number of scenarios, workflow organization delivers unbeatable performance. Unfortunately, the number of scenarios is limited to very special cases where the means of a global modelling and the means of globally modelled agility can be coped with at a reasonably high complexity. In 98% of the typical office processes, however, the varieties and “Ways to Play” require numerous interactions with colleagues and external parties, indicating adequate adaptations. Just as for games of chess, pre-modelling of the “Ways-to-Play” or business processes, no matter how good modelling languages or methodologies to agilize the global models are, have practically failed in most real life office scenarios.

Process management here takes a fairly different approach as it neither tries to support project or target domain decompositions nor the implementation of global processes. Process management rather supports the development of a response process to a stimulus with a focus on the chaining of well-defined, proven and to be executable blocks that carry the enactment of the value added to be delivered by a business organization. It focuses on modelling work packages and modelling how these work packages can be linked with each other to assemble a suitable process. In the industry we have learnt about the necessity to balance flexibility and rigidity, we have learnt that thinking in global models, states and state transitions, global rule engines, global models of knowledge representation, etc. are leading to unhandeable complexity and need to be avoided. Focusing on a local modelling and introducing a set of methodologies to pragmatically realize the chaining, or as it is proposed to be implemented here, the **forward propagation**, is key to success.

The next-generation process management tool, as described here, is supposed to replace an existing process management tool in industrial application at Einhell today, spanning over several locations worldwide, involving several dozens of teams and several hundreds of staff. The existing solution is

carrying our product development organization as well as our quality assurance organization, our product sourcing, supplier engineering and marketing efforts. The existing solution is limited as it is built into our ERP systems and has to cope with the limitations this environment implements. The technology stack is far outdated. The user experience from another decade. The next generation PM Tool should separate the process management from the ERP system, giving us more freedom to update and changeover our ERP landscape. Moreover should the new PM Tool overcome some of the functional deficiencies that the existing tool has inherited from the environment it runs on.

This document is our ***User Spec***, describing our functional requirements and the principle conceptual approach.

Alongside this document, there exists a user storyboard as well as a few other sidelining documents that have been machine-translated and added in the second part of this document. The first part of this document shall take a rather technical perspective on the system to be implemented. It does draft some of the more important architectural elements as well as the core principals whereas the detailed architectural definition and **detailed specification** for implementation still is to be worked out with the partner to be contracted in a first project phase.

Guiding principle

The main principle of PM-Tool modeling is local modeling. This means that modeling never considers the entire process, but only the respective / the current process step and its immediate successor(s). You never know about the entire process, since it is assumed that it can never be modeled ex ante.

The processes are assembled out of PBBs. The assembly takes place with following mechanisms:

- Direct linkage
    - conditional linking
    - fixed linking
- Selection by user
    - supported by a suggestion system (e.g. AI, rating functions of PBBs, etc.)
    - without support by explicit search for PBBs
    - fixed selection of possible PBBs
- Automated linking
    - PM-Tool itself determines next suitable PBB (search based, AI-based)

The resulting process consists of instances of PBBs controlled by an Enactment-Engine. Each user has its own task list and the Enactment-Engine creates a task for a user to get to the corresponding PBB and work on it.

In addition, the software must have its own data storage (process + domain data model).

The new PM Tool as a cloud based application:

The existing PM Tool is in operation on three continents, multiple locations thereon and spread over multiple organizational units. Today we suffer from the extensive need of synchronization between databases that host a system in several locations. A new system should from the very beginning on, be designed as a cloud based system, eliminating design considerations taking care of hosting and synchronization questions, load balancing and scalability. From the beginning on, however, we need to consider that we do have a proper access and a proper handling performance in all the main Einhell countries where the system is applied. This includes all European countries, mainland China, Hong Kong, Australia and New Zealand, South America, Middle America, North America and South Africa.

The system should be easily accessible by the use of all Windows based computer systems ideally running inside a browser window. In addition, the system should be able to be accessible and useable through mobile applications on Android and iOS machines.

User Story: Explanation of the key terms PBB, task, capability, work step

Alongside this document we will use the terms PBB, tasks, capability, and work step at different contexts, in order to straighten out our intentions, we would like to give the following explanation:

- PBBs: are computational entities that encapsulates computational functionalities needed for chaining, process control, process execution and the like. In this regard, they are a part of the system's coded object structure (IT system). PBBs are modeling elements and thus are parts of process models to be realized within the IT system to be developed. Conceptually, they represent work packages in the "real world" that physically need to be enacted while operating a business process. They throw tasks to users that are intended to enact the work package and they rely on users having the capabilities to carry out the work package.

- Work step: The work step (or work package) is what needs to be enacted in the "real world" throughout business operations. Work packages represent the granules of a business process in real life. They are the units of work employees, partners, etc. carry out along business processes.

- Task: Tasks are scheduled for users of the system to indicate the need to process a certain PBB and execute the work package associated with it. A single PBB might trigger not only one task but rather trigger a task multiple times or trigger several tasks at different phases of the PBB's enactment (initial work phase, review results from subsequent PBBs etc.).

- Capability: The capability is what users bring in to a business organization in terms of personal skills, professional skills, etc. In order to be able to properly enact a PBB with its assigned work packages a user will need a certain capability. In the context of the IT system to be developed the capability again shall be understood as a computational model entity that is used to implement a systematic capability management. As described later, these entities should model training needs, document user experience, etc.

PBB (Process Building Block)

A PBB represents a specific capability of users. The PBB represents a specific work step in a process. PBBs cannot be nested into each other (no Macro-Micro-Level breach!). This means that a PBB itself contains all the information necessary for completing the work step. PBBs are linked with each other but during the execution of a process, a PBB can only have one direct predecessor.

A PBB itself must be versioned. A version of a PBB describes its static state at a certain point of time. This is necessary for the correct linking execution of a process at execution time. Not all changes of a PBB are required to result in a new version, e.g. an update of a label because of a typo should not result in a new version of a PBB but can if the person responsible for the PBB decides so. The versioning mechanism should support roll-out of changes for a) only new or b) running projects, where possible/suitable/reasonable.

Every PBB has a **primary and a secondary owner**. Those owners are professionally* responsible for the PBB and define how the PBB works from a professional point of view. The owners of a PBB can create / modify it or delegate the work to a dedicated set of **PBB maintainers.**

(*"professionally" in this context shall be understood as the functional and professional responsibility of the *purpose*, *intention*, and *execution* of the work package assigned to a PBB. This includes responsibility and the definition of SOPs, verdict criteria for the evaluation of the quality of an executed work step as well as the relationship and interaction with relating work steps in an operational process.)

Structure of a PBB

A PBB consists of the following elements:

- Macro level
- Micro level
- Content Environment (see Data models)
- Inbound conditions
- Gates
- PBB status model

Macro level

On the macro level, meta-information of the PBB is stored that is derived and edited during the design of the PBB. The macro level only contains structural information, which is not intended to be modified during the execution of the process. This information includes:

- name of the PBB
- description of the PBB (rich text)

- primary owner and secondary owner [-- Explanation roles, groups… maintainability!]
- planned timeframes
- Search identifiers (tags, keys, organizationed Domain identifiers, …)
- Capabilities (see section “Capabilities” for a more detailed description of the capabilities)

In the past, for these entities the term “wrapping information” was coined and will still be found in Part 2 of the documentation. These entities will need to be derived and implemented in a first step when a new PBB is created. Later on, these entities need to be edited in a managed way by an editing environment that allows macro-level meta information to be systematically altered, edited and further developed throughout the further evolvement of business processes and building blocks. This might include e.g. updating of names or descriptions, reassignment of owners, editing of search identifiers, etc.

In particular, for tagging purposes the system should provide possibilities to maintain a tagging ontology ensuring consistency in naming (naming conventions, taxonomy, etc.), for example: rename, add, merge, split, delete, changes in hierarchy.

The **name** of a PBB is used as a first human readable identification, next to a primary key that may identify a PBB in the system databases. Whereas the **description** just textually describes the PBB. Moreover, on the Macro level the **primary** and **secondary owner** have to be stored. For maintainability reasons owners shall be considered in the system’s roles and group models so that a change or a human resource can be easily handled and does not require owner tags to be mass-edited.

It must be possible to store **planned** time specifications for each PBB. Those are resource and process time. **Resource time** is the time that the processor of the PBB actively requires for processing at the micro level. The **process time** is the time between the start and the finished time mark in the Enactment. The system shall adequately support the user by its time-keeping activities with proper functionalities to manually as well as (semi-) automatically track and record times.

It must be possible to store planned cost specifications for each PBB. It is intended that external or internal costs not directly related to the involvement of resource times can be directly considered and maintained here. Resource related costs should be calculated by resource times and hour rates for the intended resources.

During enactment for a specific project where the PBB is invoked it must be possible to document **actual resource and process times**. It also should be possible to obtain resource costs computed by provided costs-per-minute for the resources that have been engaged to fulfill the work package assigned with the PBB. During enactment time, it also needs to be possible to record actual non-resource related costs. Key Performance Indicators, such as **Efficiency**, **Utilization** and **Performance**

must be made available on PBB Level, on Project Level or on Resource/Org Chart Unit basis as suitable for the KPI.

In order to be able to find created PBBs, each PBB must have **search identifiers**, which should act like tags. Search identifiers are primarily used to link the PBB in a given project context to process execution. Under the linking scenarios “selection by user” and “automated linking” search identifiers can be used by logical search phrases to create proposal lists, shorten proposal lists, sort proposal lists, etc.

Search identifiers will include tags describing operational targets of a PBB, the organizational domain for which the PBB is intended, process types for which the PBB is intended, information or content to be derived during PBB execution for the project’s content environments.

The PBB’s macro level, from a conceptual point of view, describes how the process building block and with-it its signed work package integrate in the organizations macroscopic domain. The organizations macroscopic domain reflects all those elements of the organizational world that defines and describes how processes are operated, how work packages are sequenced or processed in parallel, or in more general terms how the course of cause and effect propagates in order to form organizational processes. In the macroscopic domain, we exclusively look at the process control layer and the mechanism how process building blocks are assembled to processes. In the macroscopic domain, we are not talking about work execution or how a certain work package is to be fulfilled by a resource working on it.

Micro level

On the micro level it is defined what the processor of the PBB has to do (work information definition). This depends on the current context of the process execution.

Next to the actual definition of the work, it must also be possible to define separately what has to be done once the PBB got terminated / cancelled (see PBB status model & Error handling) and the PBB was already processed. In case of such a clean-up scenario, following error-handling strategies must be considered:

- do nothing (default)
- compensation action (additional work information definition in the micro level)
- delegate to other PBB

Work information definition

The following structural elements are required to define the work information:

- Text information (SOP (Standard Operational Procedure) / Work instruction)

- Attachments (e.g. documents, images)
- Configurable content panes consisting of labels and Content Bricks (Content Bricks represent the objects that are built up in an ontology/taxonomy, used by an organization to collect, store and present information content that is gathered, derived and used in given projects and their related processes /PBBs; (see data models).
- Configurable checklists based on static labels or Content Bricks (supporting work packages best guided by a checklist approach) - to complete the work package every element in the checklist is to be provided / ticked off.
- Interactive forms (forms provided by IT Department)
- micro-workflows ideally integrated as a plug-in solution, building on existing, simple to use work-flow systems; as within a PBB work packages have to be properly scoped, we are talking about the need for micro workflows only (One-Person workflow) making a light weighted and simple solution workable.
- If a plug-in solution is not doable, work flows could be pre-programmed single solutions for specific PBBs to be provided by IT-programmers (possibly a first step)
- interactions with external systems
- Plugins for macro recorders / Office Robotic Systems

These elements can reoccur several times within one corresponding micro level.

For each property value of a Content-Brick the source of the data has to be defined. The following sources are to be considered:

- user generated
    - entered by a user
    - selectable for predefined (object) list
    - selectable with defined external source
- predefined values
- domain content data models
- process content data model
- external source

Interactive forms, micro-workflows (as long as not realized as a plug-in-solution with a 3rd party's product) and interactions with external systems are pre-defined and programmed functionalities, which have to be developed separately.

The micro level of the PBB conceptually relates it to the microscopic domain of the organization. The microscopic domain conceptually speaking presents those parts of the organizational world that are related to actual work execution by the resources of an organization. The microscopic domain describes, guides and supports employees when carrying out their work to fulfill a work package in its related PBB. Here we are not describing the interaction of work packages or how they are sequenced with the exception that data and information (project content) needs to be properly presented for

work execution and needs to be properly acquired and documented during work execution. This data and information thus typically is related to preceding PBBs and/or to PBBs that still follow.

The inner part of a PBB to this point is related to a different domain. Nesting PBB's thus would break domain levels, which is not foreseen at the moment. Together with some new conceptual approaches this eventually could be something to think about for future developments.

Inbound conditions

Inbound conditions represent the input conditions, which must be met before the PBB can be started. The conditions check the data in the process data model or domain data models and may or may not be prepared for a given PBB.

Inbound conditions in a sense correspond to the use of search identifiers in that they are used to realize a linking of PBBs.

In any cases where a PBB has been found by a search or is part of a list of options, the PBBs inbound conditions shall be evaluated before it is proposed.

The inbound conditions are logical clauses that evaluate the project context, based on the values that have already been acquired in the project's content models (domain data models and process data model; see data models). To this point, no matter whether a preceding PBB has been searching or seeing a PBB as a potential successor, the PBB has a chance to independently evaluate the situation, working through logical checks or computing an applicability value. Both can be used to include or exclude a PBB in a list of selectable PBBs or to compute a ranking of selectable PBBs.

Please be aware that in older documents (see Part 2 of the documentation) *inbound conditions* were termed to be part of the *wrapping information*.

Gates

Gates are exits of a PBB and define which PBBs come into question next if needed. The amount of gates is fixed. Gates are used for the control flow of the PBBs (see Control flow).

Per gate it is defined which successor PBBs are possible. Only one successor PBB can effectively be used per gate at execution time of a process. The following possibilities are required:

- Direct linkage
    - conditional linking (multiple specific PBBs of which one is selected by evaluating conditions and logical clauses based on data in the content environment)
    - fixed linking (one specific PBB)
- Selection by user
    - supported by a suggestion system (prepared search, AI, rating functions of PBBs, etc.)

  - without support by explicit search for PBBs by the user
  - fixed selection of possible PBBs
- Automated linking (PM-Tool itself determines next suitable PBB (search based, AI-based))

Note: The **architecture of the search and selection functionality** must be modelled in a **modular way** in order to be ready for future enhancement.

If a search result contains only one result entry this entry should be selected automatically. If there is no search result, a dedicated escalation step must take place. In such case, a new task for the processor of the current PBB has to be created or - alternatively - such situation needs to be handled by the event management system (escalation).

Another escalation scenario that needs to be handled is that the user believes that a presented follow-up PBB is not suitable or a given gate should not be opened at all in a given situation. In this case, the user must be able to manually throw an event for the system's event management.

Sequencing PBBs

Each gate has a sequence number, which defines the sequence of the PBBs which come next after the current one. The sequence starts at gate 1 and the PBB behind gate 2 can only be started as soon as the PBB in gate 1 has been processed. Gates with the same sequence numbers are executed in parallel.

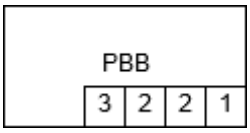


Figure 1: PBB Gates

Note: Caution has to be taken at system implementation that in cases where gates are later removed during a PBB refinement process execution is not halted by missing gates and their sequence level – this especially applies to the gates with sequence number 1 (first gate).

Process control handover

The process control handover is defined per gate. The process control handover can either be specified as friendly or unfriendly. Friendly means that the processing result of the successor PBB is of interest, this means the result of the PBB execution is to be evaluated or waited for, before the PBB can be considered to be finished (see status model) or the next gate can be triggered. Processing results of successor PBBs whose gate is defined as unfriendly are not of interest (fire & forget), this means that unfriendly triggered gates are considered "processed" as soon as they have been invoked.

Gates defined as friendly can furthermore be specified as synchronous or asynchronous. Asynchronous gates allow the following gates, even gates with higher sequence number, to be triggered without the processing of the current successor gate having returned with a result yet. The PBB itself, however, cannot be finished successfully without all asynchronous friendly gates having returned a result yet (see status model). Synchronous gates force the PBB to halt opening gates with higher sequence number until the current subsequent PBB has returned a result.

Friendly and unfriendly gate configurations can be mixed at will.

Gate conditions

For each gate it must also be possible to store conditions as to whether a gate opens or remains closed. The conditions should be defined as rules. The data for the conditions comes from the process data model or domain data models. Time constraints for opening / closing a gate should be able to be modelled, too. It is necessary to be able to add constraints that previous gates have been opened / closed.

It also must be possible to configure the maximum amount of **iterations** / retries of a successor PBB per gate. If the maximum is reached, then no more execution of the PBB behind that gate is possible anymore. To this point, at least an event should be thrown that can be handled by the event management system. Additionally, a compensation action should be able to be configured.

PBB interception

In each gate, it must be possible to configure whether the successor PBB behind the gate starts automatically or a separate ***Task*** is invoked in which the gate setting is configured (start the successor PBB behind this gate or close the gate again without starting the successor PBB). This is to skip a single gate manually by evaluating the situational context.

Another form of PBB interception is to invoke a task for the PBB as soon as the successor PBB is finished to decide whether successor PBBs with higher sequence numbers should start or not. This again needs to be configurable on gate level. This is to skip execution of one or more higher-level gates based on returned results in cases where this decision cannot be automated via conditions.

PBB repetition

To enable multiple instances of a successor PBB it must be possible to define per gate the amount of instances and whether they should get executed in parallel or in sequence. The amount of instances can either be configured with a fixed value or taken from the process data model or domain data models.

PBB status model

PBBs are required to have a separate technical state model with corresponding transitions. All PBBs shall have the same state model definition. All state transitions must have the ability to be commented if a user interaction triggered the state change. The following list shall be regarded as a list of preliminary examples – in a detailed technical specification, the state model to be implemented has to be further developed.

At least the following states are required:

- Created:
  - got initialized and added to the project's process tree model
- Waiting for conditions
  - Waiting for conditions to be met
  - Waiting for events
- Ready to process
  - PBB is ready for enactment and waits for the user's start on the microscopic level
- Planned
  - Waiting for other PBBs
- In process
  - User has started the enactment of the PBB by commencing the first task
- Suspended
  - Initiated by execution control of the system
- Postponed
  - Initiated by current user enacting the PBB
- Micro-level finished
  - Point in the micro-level execution of the PBB at which the main value added of the user to enact a PBB is done
- Finished
  - Micro-level and all friendly synchronous and asynchronous successor PBBs finished successfully
- Failed
  - PBB failed due to professional / functional reasons indicated by the situated context of the project as identified by the user or friendly successor PBB failed in situations where the failure could not be healed
- Aborted
  - PBB was "In process" but got cancelled before it was "Finished" due to systems process control
- Cancelled
  - PBB got cancelled before it was "Ready to process" due to the systems process control or manual cancellation (controlled by role privileges).

A PBB in a process is active (-> neither finished, failed, nor aborted / cancelled) until all subsequent PBBs are finished. Because of this, a separate status model for users has to be used for the ***Tasks***. Details for the task status model need to be further defined.

With every state transition, the conditions of the gate and the PBB must be checked again in order to detect process data changes automatically.

In addition to processing-related states, PBBs should be able to be assigned **static states** that depict whether or not they can already be used in processes at all (“active”) or whether they are still under construction (“draft”) or already outdated (“deleted”). There might be additional static states required to handle PBBs in libraries.

Aside from the PBB state model the process itself must inherit process states to be managed in a separate process state model (to be further defined).

PBB types

Besides the “normal” PBB which represents a capability of a user of a specific work step in a process it is necessary to have additional PBB types which provide specialized functionalities. Following PBB types are required

- start PBB
- normal PBB
- milestone PBB

Start PBB

Only PBBs of this type can start a new process for a project. Those PBBs do not require a complex Micro level but at least the mandatory Content Bricks defined for the start-nodes in process data model and assigned domain data models have to be acquired. Examples of such contents are

- Customer
- Country
- Product group

Conceptionally speaking, these contents define core data to characterize the project, define crediting data to enable the user crediting function (needed for resource asignment), define a global project priority, etc. The priority must be displayed in the task list (Task list management) and considered e.g. for the ordering of tasks.

Normal PBB

Normal PBBs will represent the bulk of all PBBs. They contain the complete Structure of a PBB as described above.

Milestone PBB

Milestone PBBs do not require a Micro level. Their main purpose is to get an overview of the process state in regards to Monitoring & Reporting functionalities. Milestone PBBs are also used for Crediting PBBs.

PBB ontology

All PBBs must be placed within an ontology. The ontology serves for grouping PBBs by business domain contexts. One PBB can have several assignments in the ontology. The ontology itself can be used to search for specific PBBs (search identifiers) when creating new PBBs or to get an overview which PBBs are assigned to different domains.

PBB simulation

It must be possible to simulate the behavior of PBBs in a test setting with predefined input sets and expected output sets. These tests primarily focus on testing PBBs in isolation (like unit-tests). Whether complete processes can be tested in a reasonable fashion has still to be discussed (see Open Points).

Control flow

The control flow of a process is determined by the defined successor PBBs in the gates. A successor PBB always reports back to the previous PBB (for unfriendly handovers, the result of the successor PBBs however is ignored). A PBB therefore remains technically open until all friendly successor PBBs whose gates have been opened are not active anymore (see PBB status model).

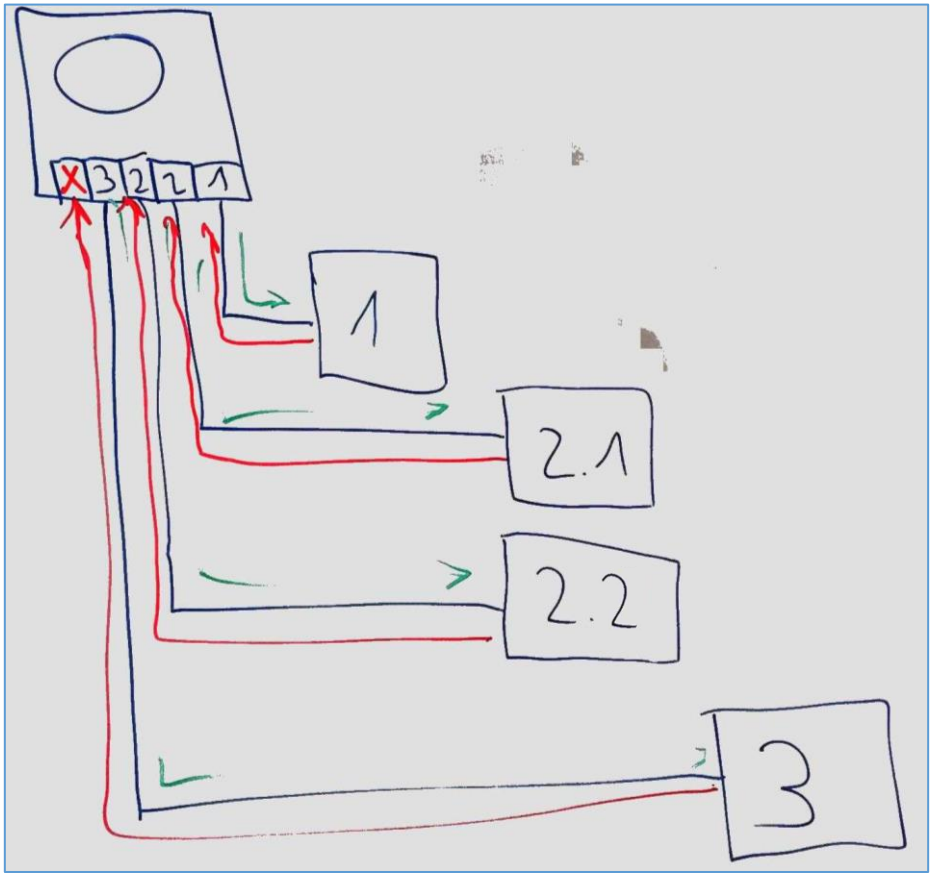


Figure 2: PBB Chaining

There are two cases for the notification (back reporting) to the predecessor PBB:

- PBB finished
- PBB failed

If a successor PBB has finished (successfully processed), the next successor PBB can continue. If no further successor PBB exists, the corresponding PBB reports finished to its predecessor.

Process execution context in a project

PBBs linked to a process of a given project share a common data space in which they store all their data for the corresponding project (the process data model). Every PBB involved within the process creates a separate execution space, which serves as a separate scope for the stored process data (see

Data models). If a PBB is aborted or terminated (see PBB status model) and afterwards executed again, it creates another separate scope and it should be possible to view / copy the data from the parallel scope.

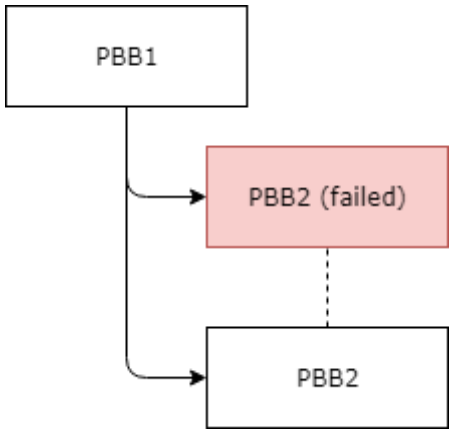


Figure 3: PBB - Execution Context

Error handling

If a successor PBB fails (triggered by user in micro level) a failed notification along with a message entered by the user of the successor PBB is sent to the predecessor PBB. The user of the predecessor PBB can then decide

- report failed to its predecessor PBB along with a message
- retry the PBB – if maximum amount of iterations is not reached
- choose another PBB for execution – if this is possible within the gate
- suspend all successor PBBs (interim solution – see PBB status model)

When a PBB fails, it has to change the state of processed successor PBBs to aborted and the state to all other successor PBBs to failed. Currently processing successor PBBs have to stop (tasks are removed from task lists, events thrown that tasks in execution are notified for cancellation, etc.). If the user of a predecessor PBB decides to retry the PBB or to choose another PBB (provided the gate was defined accordingly) all successor PBBs have to start their work again from scratch but could view / copy the data of the parallel failed execution context (see enactment environment – copy/paste functions). In case of a failed PBB execution, corresponding compensation actions defined in the microscopic level of processed successor PBBs have to be triggered (Micro level).

Events

(→ see Open Points)

User Story: “Static and dynamic process flows”

From a user perspective, the aspects of static (direct linkage) and dynamic processing (selection by user or automated linking) approaches depend very much on the functional domain, for which business processes are to be implemented. For quality assurance reason on one hand and flexibility reasons on the other hand, the right balance between a static processing approach (where exits of one PBB are statically linked to a second PBB) and the dynamic processing approach (where the selection of follow-up PBBs behind a potentially dynamically opened exit is dynamically searched, identified and selected) is very delicate. In most scenarios, the number of static links will greatly outnumber dynamic links. In particular, every project will have a rigid frame structure, depicting rigidly linked project phases that do span up the general framework for a project. Again in the next lower levels most milestone PBBs and procedural PBBs will be integrated by the use of fixed links.

Dynamic links come into play further down the roads. Where it comes to detailed enactment scenarios, PBBs more and more have to be selected situationally. Even they are small in numbers and are deeper down to the leafs of the process trees, they make a big difference in results.

A well thought rigid frame structure of the processes that we enact for our industrial projects, provide the basis for two very important functionalities required to make process management successful:

a) Simulation views and graphical representations of rigidly linked PBBs: Pulling on an entrance node (start PBB) to a project template should enable us to review the full structure of all rigidly connected PBBs to this entrance node. These are so to say the rigid skeletons in a dynamic processing world. When thinking of a new business process, the process designer in a first stage drafts a skeleton. He then implements the skeleton by implementing the PBBs and their rigid links. The PM Tool must have a possibility to preview or graphically represent rigidly linked PBBs as this is the easiest way for the process designer to see that the way he modeled PBBs and linked them up, represents his initial idea of the business process. Wherever dynamic links would dock in dynamically selected PBBs they cannot be displayed in the process skeleton, however should the onset points of dynamic links be clearly marked in graphical representations of the project skeleton.

b) Project scheduling: When a new project is triggered, its skeleton can immediately be built-up and prepared in the process data model. All milestones that are part of the skeleton thereafter can be used to calculate the project schedule based on the process time foreseen/stored in the PBBs expected processing time property. In the industry it is typically enough to limit scheduling to the skeleton level. In the foreseen process times the variability due to dynamically added PBBs is just empirically considered. Based on this, the PM Tool has to implement at least two

simple scheduling methodologies. Forward Scheduling (which schedules the skeleton nodes according to their sequential or parallel operational sequence, from a *start date* to a *finish date*) and Backward Scheduling (where the skeleton nodes are scheduled under consideration of their operational sequencing from a *target finish date* back to all the required *start dates*). Load considerations on different teams in the organization in a first step do not need to be considered. In a second step, load considerations have to be considered just by expending or shortening the process times by considering expected resource loads in the different organizational units (organigram) and by considering the assigned priority of a project.

Dynamic links provide an enormous conceptual power to process designers. The difficulty for them, however, is the prediction of system behavior at run time. It will be soon impossible to simulate all possible PBB selections opened and closed gates, etc. To support the process designer it should be possible to carry out local simulations by providing simulation data sets, for example copied from real life process and domain data models, which are then altered for the need of the simulation and by selecting a PBB from which on different scenarios would lead to different dynamic selections of PBBs. With these means the process designer would be supported in debugging configured dynamic links under different contextual scenarios, starting from a local PBB onwards to reduce dynamic complexity.

Note on static links and skeleton:

Already the existing PM Tool today creates a graphical tree representation of the skeleton. Here branches behind only conditionally open exits are still considered to be part of the skeleton. For the development of the new PM Tool, we would suggest a similar approach, regarding statically linked PBBs behind a potentially conditional exit point to be part of the skeleton. In practical applications it turned out that many of the conditions of conditional exit points can already be evaluated in very early stages of the project. Shortly after project instantiation thus, the skeleton can be refined to enhance scheduling calculation.

Data models

Alongside business processes, the acquisition of data and information is a key point. In most work packages employees carry out value added steps by working with information and by generating new information. Even when measures are triggered, non-information related work is carried out, etc., it is worth noting down what has been done and when it has been done. The process management tool thus needs its own infrastructure, structures to acquire and persist data and information in own databases in the one or the other form. They have to be an essential part of the PM tool's architecture! Under this section, we describe the data models that we would like to apply, providing the basis for thorough architectural concepts to be developed and implemented. On the following pages, we provide a technical description of what shall be implemented. In addition to this, we will add a user story, describing the functional requirements from the perspective of system users and organizational developers.

The PM Tool requires several data models on which the PBBs can operate

- Domain data models (multiple) e.g.
    - Generic product model
    - Purchase BOM
    - Safety qualification scheme
    - Function and handling qualification scheme

- Process data model

The process data model as well as the domain data models must further be split up into

- Structure data
- Content data

The data structure of all types of data models must be directed acyclic graphs (rooted trees). This kind of data will be unstructured / not stable. The data models should be modelled in a way that users can access (address the data for read and write operations) in the same fashion.

Domain data structure model

The domain structure data model consists of multiple different data buckets, which build up on each other.

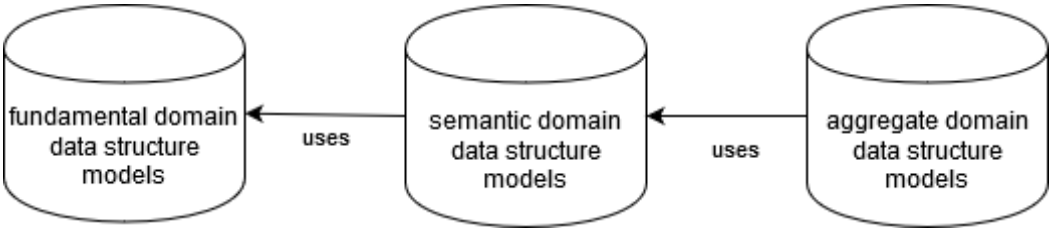


Figure 4 Domain data structure model

All domain data structure models have in common that they should be organized in a library / an ontology and searchable by tags. Each domain data structure model can have multiple tags.

Fundamental domain data structure model

The fundamental domain data structure model represents data structures created by IT developers and cannot be modified by users in any way. Those data structures represent very basic data structures like a key value pair data structure. Moreover, in this data bucket data structures from 3rd party applications can be made available (see Data sources).

User created domain data structure models (Content-Brick)

All domain data structure models which are created by users (Semantic domain data structure model, Aggregate domain data structure model, Domain template data model) are required to have specific modeling functionalities which differentiate them from the Fundamental domain data structure model and must reuse the OOP Mixin concept. The term for this modelling concept is called "**Content-Brick**".

Content-Bricks can be composed of other Content-Bricks from lower or equal domain data structure model tiers. The composition of the data structure is required to enhance reusability of domain data structure models.

Content-Bricks have properties coming from the Fundamental domain data structure model or other Content-Bricks. Properties in a Content-Brick must be addressable with XPath (XML Path Language) e.g.:

- /A/B
- A//B/*[1]
- child::A/descendant-or-self::node()/child::B/child::*[position()=1]

A property is in its simplest form a "key" and a "value" but can also be a more complex object structure like a list of values or nested / compound properties. It must be possible to rename properties (i.e. the label of the key) and provide localization.

Each property must be aware of the data type of the value and provide corresponding additional options like:

- Predefining constant values
- Constraints on data types (ranges for character, numeric, date and time types)
- Validation rules (Regular expressions and embedded script language)
- Settings for visual formatting

Additionally, it must be possible to define computed properties in a Content-Brick. Computed properties are properties which have one or more other properties (this also includes other computed properties) as value source. The resulting value (which is a read-only value) is computed based on the source value with an embedded script language. The datatype of the resulting value must conform to the result value of the embedded script language and must not be ambiguous.

Semantic domain data structure model

A dedicated set of users has access to the Fundamental domain data structure model and can create domain data structure models with semantic meaning. This domain data model bucket is completely in the responsibility of the set of dedicated users within the PM Tool. All other domain data structure models are built on top of that and must not have direct access to the Fundamental domain data structure model.

Example:

Fundamental data structure model “KVP” is used to create the semantic domain data structure model “Voltage”. Following example definition should be possible (KVP fundamental data structure must provide this functionality):

- Name
- Tags
- Value
- Datatype of value
- Unit
- Min / Max value
- Tolerance type (percentage / absolute)
- Tolerance range

Aggregate domain data structure model

A dedicated set of users can create aggregate domain data structure models which are based upon the Semantic domain data structure model. An aggregate domain data structure model can be built with multiple semantic domain data structure models.

Domain template data model

The Domain template data model serves as template for the knowledge structure (hierarchical semantic structure within which data is organized) in the application domain within which managed

processes should be applied by the PM Tool. The knowledge structure can be created by a dedicated set of users. It is based on the Aggregate domain data structure model and Semantic domain data structure model. A Domain template data model can be built with multiple domain or semantic domain data structure models.

Also on the Domain template data model individual users are granted access on whether they can access the Domain template data model. This only controls whether the user can use the data structure. This does not imply that the user has a PBB suitable for accessing the data structure (see Open Points).

Domain content data model

The domain content data is used within a process and contains all the entered values via PBBs by users within a specific process. The structure of the domain content data models are based upon the selected Domain template data models (prepopulated) which were selected/assigned at the start of a process or later on. The structure can differ in later stages of the project from the original structure of the Domain template data models because a dedicated set of users changed the structure to comply with the factual project content/physical knowledge structure and its representation requirements of the current project. In addition, it must be possible to grant access to users, which were not foreseen at creation of the Domain template data model.

Process content data model

The process content data model is used to store all the information derived in the process work packages (that might already have been stored in the domain data models – potentially as reference only) and any other information relevant to the process progression. Moreover, the following data has to be stored in the process content data model:

- Actual times (resource times, process times)
- Processor of the PBB
- State of all PBBs
- Process documentation
- Internal & external monetary costs / expanses

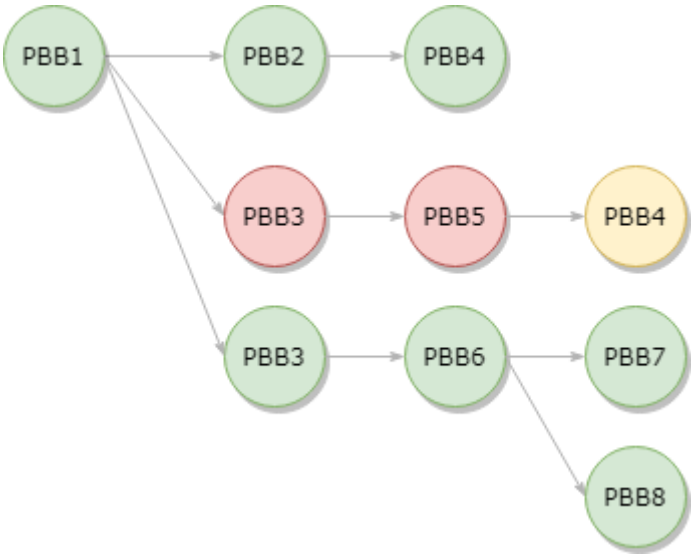


Figure 5 Example progress progression

In the figure above the color green means PBB was processed successfully. Color red is PBB was faulty. Color yellow means not processed.

User crediting data model

In the PM Tool that is in operation at Einhell today, users / resources are assigned to project roles, based on information that is acquired in very early steps when a project is initiated. For example the type of a product that is to be developed in a development project, leads to the assignment of relevant project engineers, being expert for the given product type. The existing approach in a sense is rigid and requires manual interaction when it comes to, for example, inavailabilities of certain project resources. The new PM Tool must be able to handle automated resource assignments to PPBs and with it provide enough agility at a given situational context. On the PBB it should be possible to define how suitable resources shall be selected. It is proposed to implement this by allowing the PBB creator to define a crediting function that computes the suitability of an employee / resource by evaluating the situational project context (data and information, collected in the data models) deriving a score for all available / considerable resources. Based on the score, the most suitable resource shall be assigned. In a practical implementation this could be done as follows:

To assign Tasks to a user a User crediting function is used. The data model for user crediting should consist of multiple decision tables (like DMN – Decision model notation) which is used by the crediting function to determine a score.

	User	Customer	Score
		"Cust1","Cust2","Cust3"	
1	User1	"Cust1"	10
2	User1	"Cust2"	-100
3	User2	"Cust1","Cust3"	8
4	User3	-	5

The figure above represents such a decision table. In following examples, the process has a specific variable for the customer set and all users in the decision table are allowed to work on the process:

- Customer “Cust1”
    1. User1: 10
    2. User2: 8
    3. User3: 5
- Customer “Cust2”:
    1. User1: -100
    2. User3: 5
- Customer “Cust3”
    1. User2: 8
    2. User3: 5

In the decision table, it must be possible to access data from the process model as well as some specific values like “Amount of completed PBBs”. “Amount of completed PBBs” is needed to prefer users, which already worked on PBBs within a process.

While modelling a Start PBB, the columns of a decision table can be referenced so that at the start of a process the values can be filled accordingly. The reference must consider the datatype. Moreover, if the referenced column represents a list of possible values those must be selectable during the process execution.

Data sources

The data for the data model is either created within the PM-Tool itself or taken from ext. sources in form of third party applications.

Third party applications have to be connected with the PM-Tool by a separate layer for reading and writing the data. This layer has to be developed in a separate program (see Open Points) and does not have to be included in the first version of the PM-Tool. The data structures of the third party applications have to be added to the Fundamental domain data structure model.

Possible data exchange definitions with 3rd party applications must be defined ad hoc for the specific context. This is based on the fact that the time of synchronization can't be determined ex ante. Additionally, the lifecycle of the other application must be integrated with the dynamic data structures of the PM-Tool.

Externalizing the data exchange in a separate layer (i.e. another dedicated application) ensures the most flexible way to synchronize the data with other applications.

User Story: “Data Models / Content Environment”

The acquisition of data and information as well as the access to data and information is critical for the execution of business processes. To support the dynamic processing, the PM tool has to cater for adequate infrastructure to structure, to store and persist as well as to retrieve and present data and information for the use in the microscopic work packages (presented to users during enactment) as well as for process control decisions. In addition data structures should be accessible for control purposes, for enactment purposes (for example the assignment of resources), for reporting purposes as well as for measuring organizational performance and suitable KPIs, balance score cards, etc. later on.

Collecting or presenting flat data in flat lists would be unsuitable for the purpose intended here. Every bit of information, every data acquired or stored, needs to be properly contextualized in order for users / resources working on building blocks and their related work packages to properly understand and work on them. The data models or the content environment discussed, thus has to be able to model both, structure (for example trees) and content (for example content bricks). Both together form suitable data models, of which we require not only one, but rather at least two or even multiple to be assigned to a single project.

It is obvious, that the new PM Tool has to implement its own data and information holding structures, based on its own data base systems, suitable to realize required data models. These tools in particular have to support the implementation of generic structures and generic content items that support the development of a suitable language space for our employees in their professional working domain. In addition, we have to have a generic language space, related to the business processes or processing in general, depicting when and which process building blocks have been enacted and in which relationship they stood as well as which contents have been acquired in which work package.

Thus, we propose to think on the roughest level when it comes to data structures in so called data models. They define the ways we structure and store as well as we retrieve and present data. Throughout our discussions and with an eye on the existing implementation at Einhell, we propose at least the two following data models:

- Process data model
- Domain data models (multiple!)

Each of the data models is having its structural set-up of nodes and edges (trees) its inner coherent structure along which data can be retrieved, queries can be executed or data can be persisted – everything in a fairly simple way, that allows configurators to be built to easily define a data retrieval, a query or the location into which acquired contents have to be persisted. As mentioned above, we believe that tree structures would be most suitable for process data models as well as domain data

models. Most users implicitly understand tree based data storage approaches as in use by file systems since decades. They already have been implemented in the existing process management tools at Einhell and turned out to be suitable. From this perspective, we should consider graph based data base systems as potentially most suitable choice. They offer a particular simple access for data extraction as well as data persistence in tree-structured models. However, it is important to well consider the existing deficiencies that graph data base systems still show in real life applications up until today.

The process data model is oriented in its basic structural principle at the evolvement of the business processes. Nodes and edges are generated alongside with the invocation of process building blocks (PBBs). Every PBB is a node in the data model. Every edge links gates (exits) of an invoking PBB to the invoked PBB. The generic structure of the process data model thus naturally develops alongside the enactment of a dynamic process. All contents derived in the work packages assigned to a PBB, are stored / persisted right with the PBB's representing node in the process data model.

The domain data model, assigned to a project or the multiple domain data models, assigned to a project have a different structural orientation. Where their tree-structured hierarchy is rather defined by the needs of the professional domain in which the project is executed. In projects for example, that are targeting the new development of a machine of a certain type, may for example follow the principles of a product data management structure depicting functions and physical implementations of a machine or its generic machine element structure or its BOM structure. Typically, for the professional domains it is known which domain models work best to store and retrieve data and information. Such data structures could for example be loaded as templates from template libraries to support the projects (for example structural PDMS (Product Data Management System) tree libraries, holding structural trees for angle grinders, impact drills, lawn mowers, etc.).

In many cases, however, these domain data models may be further developed or modified throughout the project by adding or deleting templated structure. In different professional domains, a number of different domain data models are required. These domain data models then have to support diverse content models, modelling different types of products, services, storyboards, or other content aggregations. Along process execution, again the nodes of the prepared or developed structures will be populated with content. Aside from pure structure, templates could already contain provisions for contents that are expected for a dedicated node. In addition, even actual content could be loaded along with the templates. Instead of loading a template, an existing domain data model from a previous project could be loaded together with all the contents, previously derived as versioned data. This would for example be the case for a product improvement project, triggered to improve an existing product that has been processed in the product development team in an earlier project. Throughout the

improvement project, the existing data would be the basis for the improvement work, the domain focus would be to define the changes to the existing product and thus to the existing data structure and content to be acquired and persisted.

Where within the process data model, contents are generated and persisted that are developed alongside work steps of business processes, this is not the case for the domain data model. Data structures in the domain data models describe the relationship of data and information in relation to each other or in relation to a pre-defined professional domain model. In particular, for the content in a domain data model we have to expect that external data sources may play an important role. Contents could need to be acquired from external database systems or other data feeding applications. Just as after processing this data and information and deriving modified information, these modifications might need to be handed back to external database systems or data consuming applications for persistence. From a system theoretic perspective, thus we would like to draw the system border just alongside the database systems of the new PM Tool and its interfaces, where required, to external data sources.

The content:

Up to this point the user story has been purely addressing the structure, into which contents should be persisted in the data models. But what should be persisted here is content to be presented for processing, to be persisted after processing or to be queried, to be modified, to be acquired, etc. Content, however, is more than information and definitely more than data in the context of this system. Content rather requires itself a possibility to provide a physical structure, presentational functionality, computational functionality, etc. Content is where we want to store data in! But content is also what we want to present within the microlevel of a PBB! Content is not just numeric data sets, a string or an object reference but rather do we want to store information relating data or values to keys and physical units. Content should be something physical, an entity that we connect to for example for the microlevel of a PBB, for a control query in PBB exits or for reporting purposes etc..

For this reason we believe it is required that we define a dedicated object model for content, that provides all the classic methodologies known from object oriented modelling, such as classes, instances and heritage. With this, we aim to span up a structured library of available content objects. With the object-oriented approach it should be possible to enrich the pure data by additional data structures and functions. Instead of single data items, we are aiming for the representation of simple or later even complex information. Such as, in the simplest case, key value pairs, up to objects, that would hold multiple keys, multiple values or matured structures such as vectors, fields or other objects, units, requested values, actual values, tolerances, functional and logical evaluations, etc. Based on this we could implement typing rules (weak or strong typed data concepts), data queries, logic functions to

check consistency or validate data, persist data, etc. We should exactly here provide functions to convert physical units or present values converted correctly to user expected units under given local contexts.

We name the objects of the content models "Content Bricks".

An integral part of the content bricks object functionalities must be the possibility to access existing information in one of the data models by the definition of a simple query. During execution of a process building block, the execution of the defined query would populate the contents to be displayed during enactment. The queries have to be configuration items to be defined for example in configuration wizards when preparing a content brick for a PBB or when adding a content brick during process enactment to a PBB. The query definition (its configuration) has to address both, the source data model as well as the query pattern that reflects on keys, labels, IDs or contextual patterns, such as expected neighboring nodes.

Throughout an evolving business process execution, contents might be acquired and later on revised etc. The results of content brick queries might thus deliver multiple hits that need to be properly displayed in the user interface the content brick is implementing for its content. This could for example be done in a list sorted by youngest to oldest acquisition date.

Aside from the functionality to configure a query to populate the content brick, there needs to be a matching configurable persistence function. Content acquired during an actual enactment stage needs to be properly persisted into the data models so it can later be queried. Of course, only actual or changed data must be persisted and not all the result set that a query originally revealed.

A second important functionality content brick object have to provide is the representation of data and information in a visual graphical user interface element. In the content pane of a process building block, contents from content bricks should be plug-in elements providing the right visualization for the type of content to be displayed. As within the content brick object the data typing is realized, units of measures are defined, a necessity or existence of tolerances or tolerance bands are addressed, requested, offered, agreed or actual values of properties are defined. For an object brick the form of data, be it a single value, a vector, a matrix, a list, a key-list etc. has to be designed. It is obvious that the way this data is presented is to be defined here. It would be impossible to define data representation from the perspective of a PBB or while a PBB is conceptually drafted. Data representation has to be linked to the data to be presented.

Due to a potential complexity in data representation, the GUI has to provide reasonable functionality to support convenience and usability for our agents working with the system. At first glance, the

visualization has to summarize relevant information in a compact way. But the GUI hast to have functionalities to allow a drill down into the full data structure of a content brick. It also needs to include feasible solutions to edit or enter content in adequate ways. Additional functionalities might include suitable views that might be displayed in detached window elements providing enough space for a proper data representation. The data representation in addition does need to display the result of functional checks or logical checks initiated after data input or data changes.

After a user has been working through a PBB (work package) in project enactment having analyzed contents, changed contents or acquired new contents, the contents have to be persisted. As described above persisting data should happen in a close analogy to querying data. The difference here is that the persistence will definitely always happen in the process data model in which all activities are persisted by default. Aside from the persistence in the process data model, contents might be persisted in domain data models as well. Whereas the persistence command for the *process data model* is to be implemented as a standard, the persistence commands for storing contents to the *domain data models* need to be configured. This configuration for one or several data models has to happen ideally wizard-based when a content brick is prepared for the use with a PBB or when a content brick is added during enactment time to a PBB.

Library of Content Bricks

The object model of the content bricks requires an own library. The library is the basis for a domain ontology, helps to maintain and administer the conceptual development of the ontology and with-it the proper use of content bricks in PBBs. It should support the users by identifying suitable content bricks for a given PBB (during PBB design phase), for domain data models (during their design/development) or to situationally add bricks during enactment.

Under most normal situations, PBBs are readily prepared with their content bricks. In order to do this there would be two possible approaches: approach a) would build on the object methodology to derive a sub-class of an existing library element or a dedicated object brick, which would be a new exclusive object for a given PBB. Approach b) would reuse the defined object in the library (dedicated content bricks) and just instantiate these objects for a given PBB during enactment with a defined object parametrization. The parameters to be loaded during instantiation, e.g. from a parameter DB, would specialize a content brick for its use in the given situation, at/with the given PBB. These parameters will need to include query phrases, persistence commands, sorting identifiers to identify plug in position in the PBB GUI etc.

Whereas approach (a) leads to the buildup of a larger amount of computational classes that might need to be distributed during run time to client computers running the PM tool, approach (b) requires a

dedicated data structure for the context based instantiation of content brick objects. Beside a pure library of object classes, thus an additional database to hold context relevant parameters would need to be part of the content brick library (Parameter DB).

One computational principal of the PM Tool should be that PBBs are enacted at a given moment of time and are closed after the user has finished the enactment. At this point all persistence has to happen and all instantiated objects are closed. If it later on turns out that a given work package did not deliver the intended result or failed and needs to be repeated, in the business process a new branch is opened and the PBB is instantiated a second time. With it all content bricks are instantiated a second time. We do not expect objects who “stay alive” a substantial time during process execution. Definitely, do we not expect a re-visit of PBB instances or open content brick instances in later stages of the business process to rework them, for example in situations where the initial enactment failed! This would indicate that we do not need a versioning concept at first glance of the content brick object in the process data model. However, might data be altered in a content brick during enactment of a PBB that requires longer times or multiple revisits / tasks, thus suggesting a version concept is implicated.

The data persistence in the domain data models clearly requires a ridged versioning concept! As within the domain data models contents at a defined domain data node might change throughout the execution of a business process. For example, the power of an engine that has its representation in an PDMS domain data node might change throughout the execution of a development project from inquiry phase to design phase or qualification phase depending on the evolvement of the product development. In the domain data model thus multiple sets of data can be created and multiple versions of data sets to a content brick have to be safely persisted with a clear reference to time stamp, a link to the PBB and its position in the process data model within which the new data set has been created and the user who was working on the PBB at this point.

As described above it will be necessary to equip the content bricks themselves with additional data structures. This data structures will also help to describe, define or tag content bricks, help with identifying family relations, etc. This metadata should be part of the library and should support the users properly utilizing the right content bricks at a given situation. Typically decision on when and where to use a certain content brick can be taken in a conceptual stage. Mostly thus, this will happen under two situations:

A) when designing a new PBB or refining an existing PBB: a user with the ability to take care of PBB implementations (maintainer) will, together with an expert related to the business process (owner), define the content bricks that will be assigned to the PBB as well as define queries and persistence commands.

B) when building a domain data model the nodes of the domain data model represent knowledge aggregation levels where typically a certain content is expected. For this expected contents, content bricks should be provisioned that maybe assigned as empty containers to be later filled throughout business processes using the domain model.

In rare circumstances an employee enacting a PBB may run into a situation where a certain information is acquired that has not been foreseen in the PBB or related domain data nodes. In such situations, it should also be possible for a user to select a suitable content brick and document his findings. In such situations, the selection might be limited to a certain subset of content bricks available in a certain section of the library.

Content brick "object distribution" – architectural model

Typically, object definitions are provided as programmatic entities, compiled or to be interpreted files for the executing runtime environments. In classic systems these would be files in a files system. In the context of the PM Tool, however, we might have to handle a large number of object definitions at a given moment and a limited number would be relevant for programmatic execution. As the PM Tool should be strictly designed to run as a distributed application/cloud based application, the system architecture has to cater for efficient distribution of content brick class definitions that are situationally delivered to the user clients. We would need to understand how such architectural concepts would look like and which consequences arise from them.

Library of domain data models

As described, the PM-Tool relies on the existence of numerous domain data models suitably modeling knowledge to be built up alongside a business process addressing a certain commercial target. One or even several of these models might be assigned to a project enacted by the process management tool. As described, these domain models are to be modeled in the form of trees (graph models). Trees themselves consist of branches that can be arbitrary aggregated to larger entities finally forming the tree. To ease handling of these models in real life scenarios already the existing PM Tool holds all these tree nodes and branches in a domain model library. A base library here just holds a flat list of node objects out of which the branches can be composed. The new PM Tool should at least have the possibility to provide a structured base library that sorts the base nodes in families and allows the nodes to be contextualized with meta-information (IDs, tags, descriptions, sorting identifiers, etc.; Base Library Ontology). The most important part of the domain model library however is the library of template structures. Nodes connected to each other forming branches should be able to be stored as templates. Branches connected to each other are templates in the library on higher levels. Complete domain models consisting of an arbitrary aggregation of branches are the top-level templates that would

typically be selected at project instantiation and assigned to it. If the domain model needs to be modified throughout the execution of a project, the domain model is typically altered by deleting branches that are not needed or, under certain circumstances, by adding branches that are readily available in the template library. This requires a template library that again allows a suitable contextualization of prepared templates providing the means to define meta information such as tags, descriptions, identifiers, family relationships, etc. (Template Ontology).

Provisioning contents in domain data model's template library

As available today (the domain data model in use for today's PM Tool is the PDMS) within domain data model templates it must be possible to provision contents. For nodes in templated branches or templated domain data models, it must be possible to prepare expected content bricks. These content bricks act as identifiers that certain contents have to be acquired for dedicated modelled nodes. As content bricks are designed to be feature-rich objects, they allow to display keys or descriptions, units of measure, etc. For this particular application, the bricks, however, may be rendered empty or at least partially empty. Practically such a provisioned content brick, for example for a node called "engine" might be "power" measured in the SI unit "Watt", but in the template the actual numeric data, depicting the amount of Watt of the engine, still is left blank. Working on a project with a particular domain model assigned, provisioned contents hint for the acquisition of dedicated data and information to be acquired for the projects. Such provisioned content bricks must have the functionality to specify whether they are required to be filled out and content checked (valid logical data verification at input) or not. If they are required, missing data inside the bricks should be highlighted or indicated in data model reports. Missing data in required content bricks may also fail or prevent finishing related PBBs during process enactment or throw a related event. This is to avoid an inadequate execution of defined work packages in PBBs and to assure that data and information is adequately built up during the enactment of a project to support the execution of the related business process. Such provisioned content bricks, however, could also come with partial information or complete information. As an example, in some cases we expect certain materials to be used for certain components – for diffusion discs in Jet Pumps and House Water Works we expect and only accept metal discs. Thus, we would provision a content brick, depicting the material of the diffusion disc to be expected as "stainless steel". In such cases, during project execution, only the actual material needs to be acquired and filled in and, to this point, it then would be possible to check whether the actual value is fulfilling the expectation by a logical check (in the engineering domain this would implement so called "design guidelines" – see design guidelines in the existing PDMS System).

In other situations, content bricks for certain branches maybe completely filled out. In these situations, the provisioned content bricks provide situational contexts within a domain model. They act as a

definition of the situation, they describe for example, which conditions have to be established at PBB execution (in a domain model for example describing qualification tests to be carried out for a lab qualification project, such provisioned and completely filled out content bricks may define ambient temperatures or humidity, etc. that have to be correctly adjusted before, for example, a heat test or a durability run is to be carried out). These types of content bricks are called “definition content bricks”. With these means, the user, enacting a certain PBB does get clear instructions for the fulfillment of the related work package of the PBB, in relation to the assigned domain model. In addition, such provisioned contents can be used by the PM Tool’s process control mechanisms to control routing decisions (for example to control, whether a certain PBB gate is open or not, or to support the automatic selection of a follow-up PBB at a certain gate).

Provisioning contents should be able to happen on node level (whenever a certain node is used in any branch template or domain model, provisioned content bricks are applied), or within nodes, belonging to a particular branch template (the provisioned contents are only applied to a given node in the domain model, if the node was added by using the defined branch templates). Templates aggregated out of several branches and sub-branches aggregate provisioned content on node levels and on branch levels from their sub-branches as well as provisioned content bricks, dedicatedly defined for the aggregated template.

Integrating content bricks from domain models into PBB execution during enactment

As described earlier, content bricks are typically assigned to PBBs during the design phase of a PBB. In such situations, content bricks are selected and their queries are configured accordingly. Queries hereby can be configured to either act on the process data model or the domain data models. In a typical application, thus, provisioned content bricks in the domain data model are found from within the PBBs at enactment time by executing the queries of content bricks, provisioned in the PBBs. After query execution, these provisioned content bricks in the PBBs display the contents already existing in the domain data model. In a second scenario, the user should be pointed to select one or multiple nodes that he is working on while enacting the given PBB. While this is not necessary for all PBBs, during the design phase of a PBB it should be possible to configure a PBB with the need to link dedicated nodes from the domain data model. In later phases for such PBBs the selection of related domain data nodes could also be automated by prepared queries or node searches in assigned domain data models. In such situations, the PBBs content pane should be showing content bricks not only prepared by the PBB itself but also all those content bricks, provisioned or available in the assigned nodes of domain data models. When it comes to a point where data is persisted during the enactment of the PBB, all acquired or modified contents for content bricks assigned from dedicated nodes in domain data models, have to be persisted and versioned automatically on the related node in the domain data model. At the same time,

as the acquisition or modification of data has been happening during the enactment of a dedicated PBB inside a process tree, the persistation simultaneously has to happen in the representing node for the given PBB in the process data model.

Where data acquisitions or data modifications in content bricks are always documented with date and time stamps as well as the PBBs, within which acquisitions / modifications have been happening (path to the representating PBB node in the process data model) and the user carrying it out in the domain data models, the persistation in the process data model has to be documented with a link to the related node in the domain data model (path to the node in the domain data tree).

Implementation of domain guidelines

As already mentioned earlier, in today's systems we do have the possibility to implement so called "design guidelines" in the PDMS that today serves as the domain data model for the existing PM Tool. In a similar manner we have to provide the possibility to implement domain guidelines together with domain data model templates in the new PM Tool. Design guidelines today operate on provisioned content bricks (today only key values) just as explained above. They are again either prepared on domain nodes or on branch templates. The aggregation of guidelines happens identical to provisioned contents. Domain guidelines are today implemented by using provisioned content bricks directly (today key values). Provisioned content bricks, implementing a design guideline just have an additional set of logical checks implemented, that evaluate when new data is acquired or existing data is modified. The logical checks in the prepared set that implements a domain guideline must be able to be linked with logical operators (and, or, xor) and finally lead to a true or false result, indicating the domain guideline is *past* or *failed*. Content bricks implementing domain guidelines have to implement a small guideline state model, indicating, whether the relevant set of data is not yet available, the available set of data indicates the implemented domain guideline is passed or the available set of data indicates the implemented domain guideline is failed. The domain guideline state is a data unit made available by the content brick, that is to be displayed in the content brick's content representation (content brick GUI) as well as it is available for process control decisions when it comes to finishing a PBB, deciding, which gates of a PBB are open or deciding on follow-up PBBs to be selected for a gate.

Multiple domain data models:

It is important to understand that not only are their multiple domain data models provisioned in the domain data template library but also will there be the need that multiple of these domain data model templates are then selected and instantiated with a project. To this point a project may have 0, 1 or several domain data models assigned. The assignment of a domain data model does not necessary need

to happen at project kick off but domain data models maybe added while a project is already under enactment.

To visualize the reason for this approach let us give you some examples for domain data models: For *product development processes* typical domain data models would be product related. They would be PDMS or BOM related models along which data and information generated throughout the project is acquired. For each product group, related domain data model templates would be built up. This would be separate models for *angle grinders*, *impact drills*, *jig saws*, *lawn mowers*, *garden pumps, tillers* etc.

At the same time, however, throughout such a project purchasing information and master data needs to be acquired, too. Depending on the country, the sourcing model, the related customer, etc. separate purchasing domain model templates may be prepared. This, for example, would be *direct purchasing China*, *EC purchasing China*, *HAFE purchasing China*, *direct purchasing Vietnam*, *HAFE purchasing Vietnam*, etc.

When it comes to a product qualification this will be carried out in Einhell China lab. Already today the lab has implemented more than 4 different families of qualification schemes. The first family is safety qualification templates. These templates depict the different standards that are required to be fulfilled by a typical product class. This would for example be *standard safety for hand held power tools, standard safety for stationary tools*, *standard for garden tools* etc. The second family would be domain models for function and usability tests such as *function and usability impact drill*, *function and usability jig saw*, *function and usability lawn mower*, etc. When it comes to the enactment of a real project, the project always has the assigned process data model available to store and retrieve data. In addition, from beginning on, a certain PDMS product class template would be instantiated and assigned, lets say *PDMS impact drill*. In addition, one or several purchasing domain data models may be assigned, for example *purchasing EC China and purchasing HAFE China*. In the qualification phase it would be useful to assign several qualification domain data models such as *safety hand held power tools* and *function and usability impact drill* etc.

Merging domain data models

In the existing PM/PDMS tool we do have the possibility to merge domain data model templates before they are applied. Here currently we have the PDMS template for different product groups as mentioned above but in addition we have brand templates, country templates, etc. These secondary templates use the same base template libraries as the PDMS templates. On brand templates or country templates however provisioned content bricks for a dedicated brand or a dedicated country are collected. Before the instantiation of the domain data model the templates selected are merged meaning that the provisioned content bricks (in our case KVPs) are added to the relevant nodes from all selected

templates. In case of conflicting identical content bricks the user merging the template has to take a decision which of the content bricks is actually added to the domain data model that is to be instantiated for the project.

Content model reporting

As mentioned earlier, it is important that contents of a node of one of the content models can be adequately presented. Within a PBB, content bricks have to be compactly reported to provide a crisp overview. GUI functionalities have then to enable users to drill down and mine for in-depth contents, which comes with the need to enlarge the user interface of the related content brick. Under certain circumstances, it might be necessary to open dedicated windows to further drill in or to ease data acquisition or data modification. For a single content brick, thus several levels of detail and size of data representation have to be worked out.

Several users working with the PM Tool, team leaders, department heads or project leaders for example, have the need to look at things from a bird's eye view. They for example will have the need to obtain a summarizing report to a project/process evolvement. In such cases, they want to see data of dedicated nodes and the structure of the process data model as such. Such nodes could be milestones or other PBBs that carry dedicated markings for such purposes. Next to reported nodes and their structure (tree), there might be the need to also summarize certain contents. This should be facilitated by markings or tags that we add to content bricks so they can be selected for display in summary reports. We have to bear in mind that summary reports will have to take different perspectives. There are summary reports visualizing the state of all running projects or all finished projects or all projects in certain states of a certain domain. There are summary reports that have to visualize only a number of projects, a subset of all existing projects, for example running project relevant for a certain resource, with a certain assigned domain model, etc. Such summary reports have to visualize process flows and contents in a very compact form.

Other summary reports should visualize on a single project while still having the need for a compact visualization here already more details on contents might be suitable. Again other summary reports will need to focus more on domain related aspects, for example summarize domain contents of multiple projects by a compact visualization of their domain data models. Other summary reports might need to visualize the domain data model of a single project.

The architectural design of content brick objects and related object models need to facilitate such reportings. In addition the architectural design of the system might want to integrate out of the box reporting tools for the implementing data base systems so we can utilize the power of standardized or commercial report builders in our visualization tools, cube ware systems, etc.

In today's PM Tool, we have the possibility to create a variety of reports called "Matrix Views" that users can configure and create by themselves according to their individual specific needs. In addition they can create outputs to MS Excel in pre-defined templates that already contain advanced reporting and visualization features. Both elements are important for user acceptance and need to be part of the new PM Tool.

Content bricks and event architecture

In today's PM Tool we are missing any possibilities to utilize events in any way. This system does not have an event architecture.

The new PM Tool should, from the beginning on, be prepared for events and handling them. It has to have a suitable event management architecture. As described in the section where process building blocks and there functionalities have been introduced, we highlighted the need for a dedicated state model for PBBs. Some of these states are of a "waiting" type, dedicated for a halted execution. In such states, PBBs could be waiting for events to be triggered. In this way, the execution of branches of processes could wait for each other. In other scenarios, events could trigger the re-evaluation of PBB execution results that have already been happening previously. In again other scenarios new projects could be triggered, etc.

At this point of time, a full conceptual design of the systems event architecture has not yet been drafted. In our eyes, however the content brick object may play a very important role within such an architecture. The content brick objects could provide the necessary functionality to throw events after certain data has been acquired or existing data has been modified in a dedicated way.

Events could be thrown immediately after data acquisition or data modification or events are thrown after logical checks, similar to the concepts discussed under domain guidelines, have been evaluated and come to the result, that an event should be thrown. Both methods must be foreseen in the architecture of the system's data model design/in the content brick object model.

Process documentation

In each process, it must be possible to store process documentation. The process documentation spans files and texts / posts by users, etc.

The files and posts by users for example could be organized in a tree structure (as of today) - like a file system structure - with folders. It must be possible to grant / restrict the access to certain folders to users of the process.

Ideally, we belief that the process documentation should be organized in a document database that allows tagging, data retrieval, indexing, searching, reporting etc. This must allow documents to be retrieved along process structure or domain model structure. Such a database could also handle user posts and project community communication / meeting notes, etc.

It must be possible to create templates for the folder structure (for a file system) or database structures (for a document database) to apply them automatically at the start of a new process.

Users, Authentication and Authorization

User accounts are required to access the PM Tool. User accounts must be grouped in user groups where one user account can be in multiple user groups. Roles are applied to user groups and therefore authorization rules for the PM-Tool must be enforced on user groups.

User account

User accounts are required to access the PM Tool and represent one specific login for a human or another application. The authentication shall be done by a 3rd party in order to provide SSO capabilities. What a specific user account is able to do is managed within the PM Tool via User group and associated Role.

Information regarding the user account is split into three types:

- User can edit information itself
- Separate Role is required to edit information
- System generated information

Information a user can edit itself:

- Language
- User is temporarily unavailable

Information, which requires a separate role:

- User working hours
- User is blocked
- User is temporarily unavailable
- Qualifications
- Organigram association

System generated information:

- Last login

Organigram

Each user must have an association in an organigram. It is necessary to have multiple organigrams to represent the structure of the organization. The organigram is used by superiors to be able to see which employees are working on which projects/tasks.

User groups

User accounts are grouped into User groups. Within the PM Tool it must be possible to create User groups and assign Roles to the User groups.

Role

Roles are predefined and cannot be changed within the PM-Tool. They are used as access control to grant access to specific functionalities within the PM Tool. The access control must be additive.

Following (not yet complete) list represents required roles

- Admin
- User
- Role for modifying User role associations
- Role for blocking users
- Role for making users temporarily unavailable
- Role for creating / modifying / deleting additional skills
- Role for creating / modifying / deleting additional skill relations
- Role for creating / modifying / deleting User capability groups
- Role for modifying User capability group associations
- Role for modifying User capability associations
- Role for modifying Semantic domain data structure model
- Role for modifying Aggregate domain data structure model
- Role for modifying Domain template data model
- Role for modifying Domain content data model
- Role for designing PBBs
- Role for Additional Capability Requirements
- Role for Cross projecting
- Role for Substitution arrangement

Substitution arrangement

Each user has to define a substitute for itself. If a user becomes unavailable then the substitute is able to see all the Tasks and is able to work on those as well. The substitute “inherits” the capabilities of the user until the user is available again. This behavior must cascade. The substitute might choose not to work on the Tasks but to delegate them to another capable user (see Delegating tasks).

Capabilities

Capabilities represent a specific skill set which are required in order to be able to work on a PBB. The Enactment-Engine uses the capabilities for the Task assignment in order to determine next candidate users for the PBB during the execution of a process.

Each PBB maintains exactly one capability. One capability may have multiple versions though. For every capability version, following information is required:

- Qualifications to be obtained by user
- Constitution / definition of capabilities (which qualification entities are required for a capability)
- Qualification can expire
    - Qualification expiration in days
    - Qualification training duration

The owners of the PBB respectively a PBB maintainer manage a capability for a specific PBB. A new version of the capability must be created at their will.

Qualifications are regarded as an independent entity that can be obtained by a user through training and examination, etc. Obtained qualifications are stored on user levels. Qualifications may be expiring based on time schemes. The time schemes may be altered if PBBs with capabilities constituting out of a given qualification are enacted by the user. Capabilities require a certain set of qualifications to be obtained by a user in order to be regarded existent.

Qualification and capability maintenance interface

Qualifications and related capabilities have to be easily managed in suitable qualification libraries and capability libraries. Managing associations between capabilities and qualifications as well as managing the libraries shall be supported by an easy-to-use maintenance GUI. This has to be done by users having the role for modifying these associations (e.g. “Instructors”).

User Story: Capability Management

In an industrial organization it is vital to assign tasks to users that really know how to carry out a task and the required work packages. In the context of a process management tool, the capability management can be institutionalized. This is already today done in some parts with the existing PM Tool at Einhell.

The principle idea here is simple. Every PBB basically reflects a capability that the employees of an organization have. When a new PBB is created, it is important to decide, whether the capability for this PBB is critical in terms of *capability management* or not. If the successful enactment of the PBB and its related work package requires a dedicated knowledge or a regular training, then this will be the case. The owner of the PBB then needs to be able to mark the PBB and configure, that resources can only be assigned if they are qualified in terms of their capabilities to work on the PBB. The owner of the PBB needs to define the capability by assigning relevant qualifications needed. In case new qualifications are needed, the owner then needs to apply for a new Qualification entity to be created in the system and prepare relevant training materials that he should document with the new qualification. He is then the owner of the new qualification entity. In a next step, he needs to select resources / users to be trained for the new qualification. The training should end for the user by obtaining a qualification certificate that is recorded for the users. Users that possess all necessary qualifications (sum of qualifications = capability) are now qualified for the PBB and can be selected by the resource assignment mechanisms. The PBB owner will need to make sure enough users are available with the required set of qualifications and push for further qualification measures if required.

Depending on the type of qualification needed, the owner of the qualification entity has to take a decision, which maintenance scheme has to be applied. For some qualifications a one time training is enough. Others may need to have a re-assessment or a follow-up training within a certain period of time. Again other qualifications may be regarded as obtained if a user regularly enact PBBs with capabilities constituted out of this qualification and a last enactment is not older than a certain period of time.

The type of maintenance that the owner of the qualification entity wants to establish, has to be configured with the qualification entity, so do the necessary time schemes.

Aside from being a sole owner and having trained a certain number of users for the PBB, there should be the requirement to define a secondary owner. This should prevent that in case of organizational changes or if the original owner leaves the company, the capability management would be rendered unfunctional or PBBs are rendered orphaned.

On the resource/user level now this requires that the enactment of PBBs is to be recorded, at least for all those PBBs that are taken care of by the capability management. In addition, training times and the expiration of qualifications need to be recorded. The user needs to have a dashboard to review for which PBBs he is intended according his org chart position and which of them are governed by the capability management. The dashboard has to depict, for which PBBs he would need to have a re-assessment / re-training of certain or all necessary qualifications in order to be eligible for assignment. There he should also see those PBBs of which the qualification times will expire soon. PBB owners on the other side need a review board to identify the number of eligible users for their PBBs, identify orphaned PBBs, identify retraining requirements, etc.

Enactment-Engine

The Enactment-Engine is responsible for the Control flow of the processes. This includes the instantiation of the Domain content data model and Process content data model as well as the creation and assignment of Tasks.

From a computational point of view, we could say the enactment engine is the run time environment for the declarative processing. It runs the processes. From this perspective, it plays a central role in the application across users and resources. It implements all necessary process control mechanisms and executes them. For user that have the permission to start new projects, it provides the entrance points to kick off a new project. On the user side, it has to implement a suitable graphical user interface (GUI) to support day-to-day operation and work with the system.

The user interface has to implement **task lists** with all required functionalities make them operator-able and hand-able for users. All tasks have to be visualized tidily. Sorting functions must be available. The ability to reschedule a task as well as possibilities to re-assign tasks, forward tasks to other user resources, where allowed, should be graphically supported.

Message board functions containing important messages for a user should be available next to the task list right on the top menu of the enactment engines user interface.

Global Search functions have to provide access to projects, content models, data, etc. From here Manuals and documentations, training materials, PM-Tool Wiki etc. must be accessible, too.

Available on the top menu of the enactment engines GUI should be an **access point to projects**, their process and domain data models. Which projects are shown at a first glance should be configurable per user. **Project Blogs/Chats** should be available here. For users with more sophisticated roles a top menu should provide an access point to template libraries for domain data models, content brick libraries and PBB libraries.

Again, depending on the privilege level of the signed in user, the top level menu of the enactment engines GUI should have an access point to a set of **prepared reports** and/or **widgets** that indicate defined KPIs relevant for the user.

On lower levels, the GUIs have to enable the user to handle searches and navigate through process trees or domain data models. From there they should be able to drill down to details. On PBB levels, node levels or brick levels users should be able to **select** and **copy** required elements. For users with the right privilege level they should be able to **paste** elements into different structural positions or for example into the content pain of a PBB that they are currently working on. For such operations copy and paste methodologies have to be worked out, so that data consistency is maintained.

In a setting section, users should be able to configure their top-level menu, their reports, their widgets, etc. Resource related settings should be easily taken care of in the GUI of the enactment environment. This includes definition of substitution users, off times, holidays, etc. At this point the user should also be able to look at its associations into the organizations organigrams, the list of additional qualifications, skills, capabilities, owner ship of PBBs, etc.

Project start – Manual and Automatic Launch

Typically, a new project is initiated in a project kick-off in which some fundamental decisions have to be taken:

- In which domain are we?
- Which domain models are to be assigned?
- What is the initial set of information to be provided?

Thereafter a starting PBB is to be selected to launch the instantiation of the process that enacts the project. The described scenario today is the most common way to start a project; it is a **manual project start**.

Aside from this, certain projects with pre assigned domains, domain models, initial data sets should be able to be scheduled on calendar entries for single occurrence or under reoccurrence schemes (e.g. annual calendars, monthly calendars, weekly calendars, etc.) Preassigned starting PBBs in these cases are automatically trigger and initiate the enactment. This does automatically schedule tasks for users and drives the project. This scenario is an **automated scheduled project start**. Still these projects do need to belong to a project owner that has to be predefined during scheduling.

In a future system also **non-scheduled automated project starts** should be able to be provisioned. Such projects would be started and instantiated, meaning they would run if certain situational conditions occur, certain events are thrown, etc..

From an architectural perspective we believe that the central processing engine of the enactment environment should be designed as an structurally independent architectural entity and so does the User Interface part of the enactment environment. The first being a rather central core running the declarative processing of all worldwide processes whereas the second is rather user-oriented, applied to regional requirements, languages etc. environment running rather “decentralized” on the clients (even though clients are centrally generated in a cloud application).

Tasks Management

Each user of the PM-Tool must have its own task list showing the PBBs that they need to work on. The tasks of a user are created and assigned by the Enactment-Engine.

Tasks are always closely related to a specific PBB. In the existing PM Tool tasks and PBBs from this perspective are not even separated. For the conceptual design of a new PM Tool however it is to be considered that tasks are introduced as a separate entity. This would decouple tasks and PBBs and would allow to spurn different types of tasks under different situations during the enactment of an PBB. This would also allow independent micro state models for tasks.

Task

When working on a task users must be able to get to the Micro level of the corresponding PBB, which this task represents. Additionally, they must be able to change the state of the PBB and put successor PBBs on hold (see PBB status model). When transitioning from one state to another the user must be able to comment the state change as well as to log the required processing time. Time keeping functionality should help the users with (semi-)automated time keeping were possible.

In the micro-level of a PBB, which users reach via a task, users must be able to access the Domain content data models as well as the Process content data model to get an overview of the complete state of the process. Moreover, they need to have access to the different Process execution contexts with the ability to copy data from failed PBBs.

Additionally users should be able to view similar processes and view the data (see Cross projecting).

Task list management

The tasks within a task list must be manageable by users in a way that they can prioritize their tasks. Resubmission should also be possible.

Along with the question of a proper task list management and task sorting comes the question of a project prioritization. When kicking off a project it should be assigned with a numeric priority identifier. The larger the identifier number the more important the project. Throughout the life of a project priorities could be adopted for strategic reasons. Tasks in the task list could than inherit project priorities and could be properly sorted top down.

In later stages, priorities could be changed for different project phases only and related tasks could likewise be inheriting this prioritization from their project phases. This would allow work groups to alter priorities to better fit limitations in resources.

Due dates and open times are other important parameters that could be used for sorting functions. In later versions we should be able to agree on a **balanced score** computed from several parameters to define an optimized ranking and thus sorting of tasks in a task list.

Task assignment

The Enactment-Engine uses the authorized users of the Domain content data model of the process as starting point and checks for users with appropriate capabilities – also currently unavailable users must be considered. For each remaining User crediting function has to be applied and the user with the highest score will be assigned for the task.

User crediting function

The user crediting function computes for every user according to the Task assignment a score and assigns the Task to the user with the highest score.

The crediting function should look like

$$S(P,U) := \sum_{i}^{n} w_i d(P,U)_i$$

where

- S is the weighted score for the user
- P is the process data model
- U is user
- w is the weight for the decision table

This crediting function must be editable with an embedded script language and should look something like this:

```
1*dt[‘customer’](p,u)+ … + 0.75*dt[‘projectType’](p,0)
```

It must also be possible to store a variant of the user crediting function for the process itself, which gets preferred over the “global” user crediting function.

Cross projecting

With cross projecting different processes should be linked because they share a certain similarity or should be carried out coherently (e.g. general projects for some customer). Once two processes are linked it should be possible to view the data of the other process.

Two processes should be linked if they share a certain similarity with regards to content (customer, project, product) and there is a reasonable time span of those processes.

The linking should be done by a dedicated set of users explicitly. Linking two processes at the start of a process as well as during the process execution with another process is required. Once two processes are linked all users within that process should be able to view the data of the other process.

In a later stage of the PM-Tool, this should be supported by the tool with automatic suggestions. A similarity measuring function does not exist yet.

User Story: “Cross Projecting”

Some projects in an organization may run completely independent from any other projects. However, a number of projects maybe closely related. This could for example be several projects that an organization handles for one and the same customer, projects, that have to be finished for such a customer at the same time, projects leading to product assortment assemblies with intended identical CI, or point-of-sale presentations, etc. Sometimes, however, projects might not seem to have a relationship at first glance but they may be related based on identical technology, identical supply chain structure, etc. In all such cases, it would be helpful for the users working on PBBs within these projects to know about each other.

In a first approach, it would be suitable if related projects were just marked manually by users, identifying the similarity. In later stages, however, we could try to identify similarities based on recognition of similar project pattern, similar content pattern, time overlap, etc. This could be achieved by computing a similarity function or by applying methodologies known from machine learning, classification and artificial intelligence science.

Once-linked projects have been identified, this should be visible to the user just working on one of these projects. While working on a PBB, the documentation of related processes (process data model) should be directly, or at least, easily accessible. Contents made available at corresponding PBBs in the other projects made available or acquired, should be made easily accessible, should be able to be selected and possibly transferred (copied, etc.) to the actual PBB.

Domain data models should be easily accessible as well. If possible, linked projects should be able to be displayed in their domain data models in comparison overviews. It should be possible to register for event notifications of linked projects, so that for example the user working on one of the linked projects gets a message notification on his messaging board when a certain PBB in a linked project experiences a state change. The current PM Tool does not support cross projecting – users currently have to manually control a simultaneous enactment of related projects. The functions mentioned above should help to support users under such circumstances.

Crediting PBBs

After each milestone and after the process ended the users of the milestone PBBs and the user of the start able PBB of the process should have the ability to rank the PBBs that participated in the process.

The ranking should be made with grades (e.g. 1-7 – 4 is used as neutral element to improve statistical analysis) as well with a comment. The ranking "process" itself must not block the process flow in any way.

User Story: "Crediting PBBs"

A typical management cycle consists out of planning, execution, reviewing and revised planning for the next cycle. As the process management tool has to implement managed business processes it should inherently support management cycles. The easiest way of doing this in the context of a process management tool, is to institutionalize reviewing activities in certain PBBs. After a process is entirely completed or at least partially completed, management staff should review the results achieved in the projects. During reviews, they should rate or credit how well a certain PBB turned out to work in terms of the design of the work package (microstructure) as well as in terms of the linking results that were achieved with the PBB. Such credits have to be stored at the PBB and should be made available in reports to the PBB's owner. Reported credits and deficiencies then should help the owner to decide, which PBBs need to be revised and in what way they need to be revised etc.

Later on, acquired credits could even be used for linking decisions in dynamic links to improve follow-up PBB selections.

Monitoring & Reporting

Monitoring

Monitoring is defined as observing the process execution and is mainly concerned with the current process execution and operation of the PM-Tool. Mainly of interest is information about:

- which processes are active
- which PBBs are currently active
- which PBBs are currently waiting
- which PBBs/milestones are overdue
- which users need a training on PBBs
- which PBBs don't have a user
- etc.

This information must be presented to the user as dashboards with respect to their roles.

Notification

Users must be able to decide whether they want to get actively informed via E-Mail if certain conditions apply like:

- new Task was assigned
- PBB is overdue
- …

A supervisor must be able to set conditions concerning his employees.

Moreover, it must be possible to configure whether the notification is sent

- at once
- daily
- weekly

The Notification System could be closely linked to the system's event management.

Reporting

Reporting is concerned about business reporting functionality. Mainly of interest is information about

- Bottleneck analysis
- Process times analysis (standard deviations)
- Multiple correlation analysis (phase X of a project)
- Matrix-Views (see Open Points)
- Project – on-time rate
- Project – in-cost rate

It is necessary that the PM-Tool provides interfaces for data exports so that 3rd party analysis tools can work on the data and additional reporting can be externally arranged.

Specific PM-Tool use-cases

In this section, some important use-cases of the PM-Tool are described. They do not fit properly in one specific section or are of importance.

Modifying / repairing a process

Modifying / repairing a process is very important and must be supported by the PM-Tool. Modifying here means changing the process data and or changing the current process execution. There are mainly following reasons for this:

- process did not evolve as desired due to
  - a misconfiguration in the PBBs
  - wrong data entered in the PBBs
- a gate was opened manually

Once a process is detected which needs modification, a dedicated set of users must be able to put the current process 'on hold', so that only they can make modifications to the process.

Those users must be able to change the process data model. Moreover, they may need to cancel / abort PBBs and select a new 'current execution pointer' (CEP) for the process. This CEP will be a PBB, which was already processed. After setting the CEP it must be possible to set the state of the Gates of the PBB which is 'in progress'. Changing the CEP must cause a new Process execution context so that modifications of a process are traceable. Once all changes are made the users must be able to remove the 'on hold' status from the process.

PBB-Maintainer

The owners of a PBB will most likely not make changes to the PBB but a separate group of PBB Maintainers. Changes to a PBB must be logged in an audit trail.

Opening a gate manually

In some cases of a process execution, it might be necessary that a Gate (s. Gates) must be opened manually or prematurely in order to continue with the process or to speed it up. This requirement arises from a not optimal construction of PBBs and their Gate conditions or simply by the needs of business reality! Opening a gate manually is a very risky and dangerous operation since following gates / PBBs might not operate properly or wrong PBBs are selected in the upcoming gates. Nevertheless, this functionality has to be provided.

Since this is a very dangerous operation for the process only users with a certain role are allowed to perform this operation. In addition, before this operation is performed a dedicated warning has to be

displayed and accepted before performing this operation. Moreover, it must be clearly visible in the UI that a gate was opened manually.

Delegating tasks

A user must be able to delegate an assigned Task to another user. The following use cases are most likely

- a supervisor decides that a Task must be done by someone else
- a user can’t do that particular Task
- substitute decides to forward a Task

Open Points

1. We currently see the layer for connecting third party applications in separate programs. The reason for this is that we want to minimize downtimes of the PM-Tool concerning changes of the interfaces of third applications.
2. Discussion about Events and notifications from other PBBs from other processes still takes place. Currently, we gather examples from sample processes. Events and notifications might need to be handled per gate and also within the micro level. Reacting to events might need to trigger process data validation also. Events have to be distinguished into interrupting and non-interrupting events
3. Granting users access on the Domain template data model follows the current implementation as well as the substitution management. We still discuss about this point
4. Whether testing a complete process can be done reasonably is still point of discussion since the premise of the PM-Tool states that a process is not known ex ante. It is surely possible for mostly directly linked PBBs but this might only be the case for early stages of the project since once the processes move to a more dynamic fashion testing those might be too complex.

   It also needs to be evaluated if groups of fixed-linked PBBs should be encapsulated in reusable modules that are linked to the project templates and only included (“unfolded”) at process execution time when the process execution reaches the linking point (e.g. the beginning of a milestone). This could reduce template creation and maintenance effort, as a module only needs to be defined once but then can be reused in several project templates.

5. Example of Matrix-Views have to be provided.

User Story: “Event architecture”

From a user perspective it is difficult to hint for the design of an event architecture in particular as such an architecture does not exist in our current PM tool system. Nevertheless, we would like to provide some input, based on conceptual assumptions.

Events may arise out of multiple occasions. These occasions could be:

1. New data is acquired
2. Existing data is changed
3. New data or changed data fulfills a certain logic test or domain guideline passed/failed
4. State change of a PBB from a defined initial state to a defined new state
5. Any kind of error in the execution (e.g. query for a follow-up PBB did not resolve in a result, no user found for an assignment to a PBB, data not available for a computational evaluation, etc.)
6. Inbound conditional check failed at a selected PBB
7. The assignment of a user to a PBB is changed
8. The task is actively postponed to another date by the user
9. A scheduled PBB has reached a critical point of time in the fulfillment
10. A scheduled PBB is late in schedule
11. Process times of a PBB exceeded planned process time by a factor X
12. Resource times of a PBB exceeded planned resource times by a factor X
13. Actual costs of a PBB exceed planned costs by a factor X
14. A project is commercially cancelled
15. A project is finished
16. The main responsibility of the project has been moved from one department to another department in the organigram
17. Etc.

Some of the events have to have defined event handlers and need to be caught (for example errors and execution), others should be only available for subscription.

Events could either be handled by computational units of the systems automatically, such as: PBBs waiting in certain states, the enactment environments changing execution control based on a certain evet, etc. Others could just be leading to a notification on the user’s message board.

On the other hand could events also be handled retrospectively, for example triggering a message to the user when he is opening a certain PBB for enactment to obtain an additional information.

PART II: User Story - “Background”

1 Theoretical foundations

Prior to the conceptual design, a theoretical examination of specific topics is useful. Therefore, the following chapter deals with a more detailed understanding of the process concept, Luhmann's system theory, basics of organization, as well as business processes.

1.1 The term “process” in connection with the microscopic and macroscopic world

The first basic prerequisite for the emergence of processes is causality - also commonly known as the principle of cause and effect (Spektrum der Wissenschaft Verlagsgesellschaft mbH). These can be physical, chemical, biological or so-called natural laws. Furthermore, also defined legal or economic ones, which have an effect on the structures, the constituent physical elements or according to which they behave.

To this point this would be the *microscopic world* of processes. The *macroscopic world*, on the other hand, describes the resulting (observable) chains of action, the processes of the system, which ultimately achieve an overall effect and thus results in the system behavior.

The second crucial aspect is the time dependency. In a figurative sense, this means that processes only exist until the causal chain is completed. In economic terms, this usually corresponds to a concrete goal. If events occur in the interaction of the two components - time and causality - this is called a process. Nevertheless, processes manifest themselves not only singularly, but also recurrently (Thannhuber, p. 39).

In the field of metaphysics, one also finds another apt characterization of processes by explaining the recurring but never completely exact repeatability:

> "A human being, for example, continues in his existence as an organism because his heart beats permanently, his cells renew themselves continuously and substances are absorbed or excreted. Even a table can be traced back to the continuous activity of elementary particles that interact with each other. In both cases, the underlying processes are never exactly repeated, so that the superordinate structure gradually changes and eventually ceases to exist as a stable entity". (Schrenk, 2017, p. 158)

1.2 Fundamental Considerations on System Theory

Niklas Luhmann is still one of the most formative sociologists of the present day (Stichweh, 2012). On the basis of his theoretical works, the concept of systems will first be illustrated. Furthermore, the key

aspects of systems such as the structure, the concept of autopoiesis and the environmental idea will be examined.

1.2.1 Elements and relationships

A system contains various types of elements. However, the elements alone are not sufficient to be able to speak of a system. One of the decisive factors is the fact that the network of relationships between the elements is not arbitrary. Figuratively speaking, for example, a container full of screws, cables and belts is far from being a car.

The organization within the system is decisive: the exact, coordinated positioning and the interlinking of the individual components of the car. This is also referred to as structural conditioning, which represents the interaction of the individual elements. However, a complexity problem can arise when the maximum linking capacity of an element is reached. The system solves this by selecting the important connections and neglecting the less important ones. (Luhmann, 2015, pp. 45-47)

In general, systems can be considered very comprehensively and subsystems can be distinguished. An example from human medicine would be the human body as a whole and the heart as a subsystem.

1.2.2 Autopoiesis and environment

Natural systems follow the concept of autopoiesis. Autopoiesis comes from the ancient Greek autos, German "selbst" and poiesis "bauen". (Jan-Mathis Schnurr, p. 1). This means that natural systems are in a continuous reproduction process and thus ensure their preservation. Reproduction takes place at the micro level. In the case of the human organism, the process of cell renewal would be exemplary for such a reproductive system (Maturana and Varela, 1992, pp. 98-99). The prerequisite for a functioning autopoeitic system is the existence of natural system boundaries that ensure the function of reproducing structure and organization. In contrast to artificial systems, the choice of system boundaries is not arbitrary in autopoeitic systems.

Everything outside of a system in general is called environment, within this environment, again further (sub-) systems can be existant (Willke, 1993, p. 283). For a better understanding, a closed system and its environment is depicted as follows:

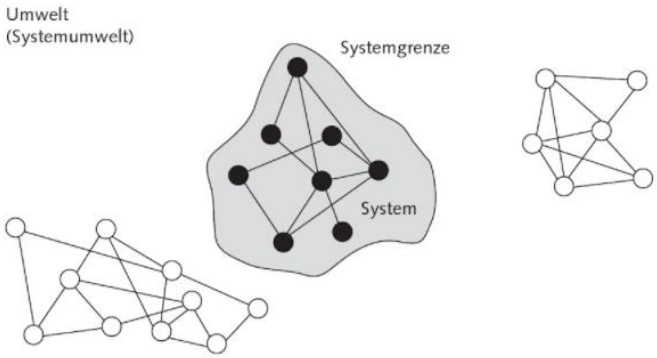


Figure 1 The closed system (Böhm and Fuchs, 2002, p. 19)

However, such strictly closed systems, as shown in Figure 1, exist only in theory. In reality an interaction and thus a networking between system and environment takes place. Those elements that create the aforementioned interactions between other systems or the environment are also called boundary elements (Böhm and Fuchs, 2002, p. 14). Element eleven (E11) from the environmental perspective and element four (E4) from the system perspective, are highlighted as examples.

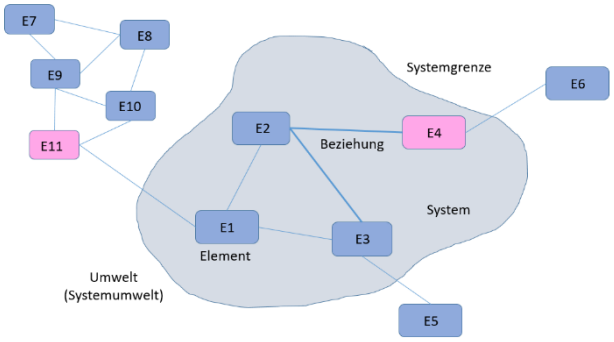


Figure 2 own adapted representation based on Böhm and Fuchs

Systems theory can be used to analyze how the system develops in relation to the environment, which in turn changes continuously. Böhm and Fuchs illustrate this changeability by presenting a system and its environment at three different points in time.

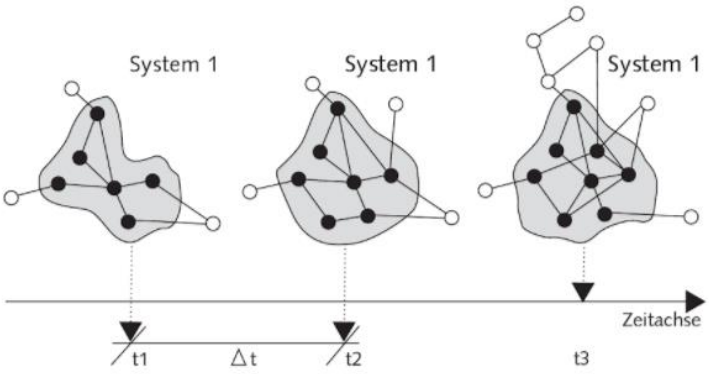


Figure 3 System state under consideration of time

It can be seen that both the links within the system have changed and the marginal elements have been rearranged from environment to system.

1.2.3 Challenges of system boundaries

Finding suitable system boundaries is a complex task. Luhmann specifies this in autopoietic systems as something independent and justifies the formation of boundary devices driven by self-selectivity of the systems themselves. (Luhmann, p. 53).

Böhm and Fuchs also describe the famous "eye of the beholder" as a factor not to be neglected. Depending on the knowledge, as well as the experience of the observer, the assignment of the elements to system and environment is purely subjective. However, this is only possible with technical systems that are delimited from the observer for the purpose of discussion.

If a far-reaching system boundary is drawn, this offers an increased possibility of the most diverse structural conditioning. At the same time, this increases the general complexity of the system. Consequently, an analysis is associated with significantly more effort. For example, investigations can be carried out based on the number of relationships of the elements within the system and with the environment. Here the principle of the overweight of the inner bond applies. According to Böhm and Fuchs, as few and simple connections as possible should be chosen to the environment (Boehm and Fuchs, 2002, pp. 14-17). Systems that are too narrowly defined, on the other hand, may refuse to have a stable system structure due to reduced internal linking possibilities.

1.3 The company as a system and organizational learning

Derived from system theory, the structure of the company describes the individual elements of which a company consists. The figure shows an example of a conceivable structure of a company and its environment.

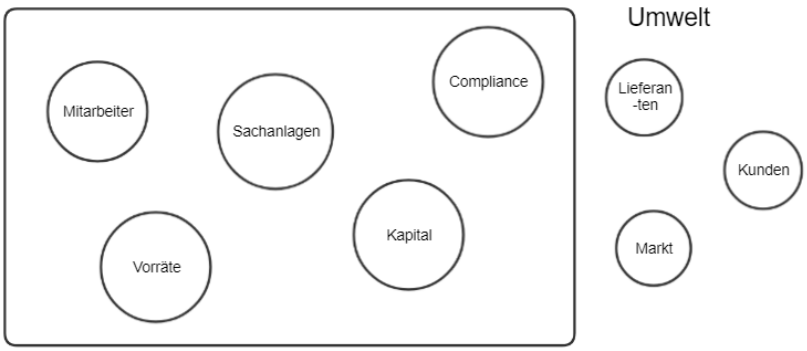


Figure 4 Own presentation based on a company balance sheet

For the sake of simplification, this is oriented towards a company balance sheet. The individual elements represent balance sheet items. In addition, the structural element “Compliance” has been added, which is a functional subsystem that is of a “permanent nature” in larger companies. Although it can be difficult to describe the boundaries of system, Luhmann achieves a system classification in his theory. Miebach illustrates this graphically in his work (see Figure 5). According to this, a company can be regarded as an economic organization, as part of the social system (Miebach, p. 33).

Figure 5 System classification (Miebach, p. 33)

The literature of Thannhuber shows a further subdivision of systems. Identical to Luhmann is the inclusion of companies as social systems. However, in his model shown in Figure 6, Thannhuber also takes into account other components such as complexity, degree of freedom and time.

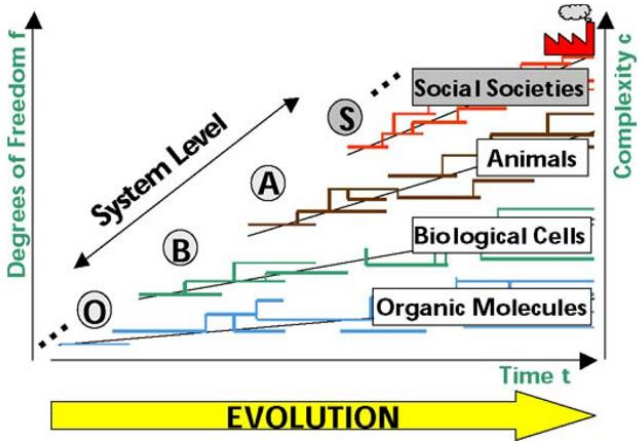


Figure 6 System levels (Thannhuber, p. 25)

This way he classifies systems into different levels. In particular, he illustrates the increasing complexity with a simultaneously increasing degree of freedom. In other words, this means that the system boundaries are being drawn more and more comprehensively, thus increasing the number of possible structural conditions. Evolution has led to the integration of systems consisting of complex constituents, which can themselves be understood as a system. Social systems, such as companies, are systems, which can be understood as integrated from at least three subordinate system levels - they are therefore level 4 systems (see Figure 6).

Processes can be viewed from two different angles: The macroscopic and the microscopic world. For companies, the macroscopic world can therefore be defined as the observable business processes, among other things. In the microscopic world, on the other hand, the action, work implementation solution seeking etc of individual employees can be found.

In addition, autopoiesis can also be seen in the system "enterprise". Structure and organization of the system Enterprize is constantly “reproduced” by the incorporation of new procedural building blocks and new links coordinating them to new processes. *Emergence* is phenomenologically the development of macroscopic behavior of a system through the coordinated execution of individual microscopic actions. This results, for example, in economic success.

Particularly successful companies, however, do achieve a kind of *evolution* during their life cycle by appropriate structural conditioning of their system – deriving successful structure and organization. This is an evolvement of collective knowledge and skills and can also be generally described as *organizational learning* (Hamella, 2018, p. 4).

For learning within an organization, the interaction of two elements is striking: On the one hand, *intelligence* at the organizational level, which requires a suitable infrastructure to generate dynamic process worlds. On the other hand, the *knowledge* that is reflected as content of the infrastructure.

The following chapter describes how action lines can be anchored in the company.

1.4 Modelling of processes in companies

First of all, it requires a visionary idea of how a business process could run. The following design parameters must be taken into account:

- Labor resource ("Who?")
- Division of labor ("What?")
- Work equipment ("With what?")
- Working method ("How?")
- Place of work ("Where?")
- Work sequence ("When?")

In the first step, a concept of action for the work resource must be drafted up, in which the exact definition of the division of labor, the work equipment, the work method and, if necessary, the location are determined (Klimmer, 2012, p. 116). The performance of the work resource can be influenced by by training. A similar approach applies to the introduction of technical equipment, for example, which becomes operational depending on suitable programming or adapted settings.

Only in the next step happens the modelling of the work packages in the form of **building blocks** that can be linked into a process. These so-called **process building blocks (PBBs)** are generic elements of the system. Both, the microscopic level (the work package, SOP, etc) as well as the macroskopic level (linking, gates, …) are actually modelled with them (Thannhuber, p. 61).

2 The current process management tool of Einhell Germany AG

Chapter 4 describes the development of the current PM-Tool. In addition, its more detailed system structure and organization is explained and illustrated by several pictures. Finally, a practical example from the field of Supply Chain Integration (SCI) is discussed in more detail.

2.1 History and fields of application

The development of the PM-Tool is based on the discussion of the current technical director Dr. Markus Thannhuber with the following problem: How can "knowledge" of continuously developing company organizations be manifested in the system? He provides the answer to this question in his work *"The Intelligent Enterprise - Theoretical Concepts and Practical Implications"* published in 2004. Thereupon he worked, together with IT specialists, on the implementation of a system-side solution. After various development phases, which will not be discussed in detail here because they go beyond the scope of the work, the current PM-Tool finally found its permanent place and could be successfully integrated into the existing ERP system *Microsoft Dynamics NAV - Navision.*

The system was first used in 2005 in the quality assurance department of the company in Shanghai. Two years later, the PM-Tool was also successfully implemented in the parent company in the areas of product design, product development and engineering. In the meantime, the system is operating in Europe, Asia and Australia. An internal evaluation (as of October 2016) shows that more than 17,000 projects have already been successfully completed in the area of quality assurance (QA) and about 11,000 projects in product development, supported by the PM-Tool. A product development project, for example, comprises the steps from technical development to marketing of a new Einhell cordless screwdriver.

2.2 System structure and organization

What does the current system architecture of the process management tool look like?

The outlined diagram is a simplified representation of system elements and system environment, as it can be found in various computer science sources in the field of software engineering (Ina Schaefer, 2011, p. 18). For reasons of simplification, the system environment in this drawing is only shown from the respective users, suppliers and customers.

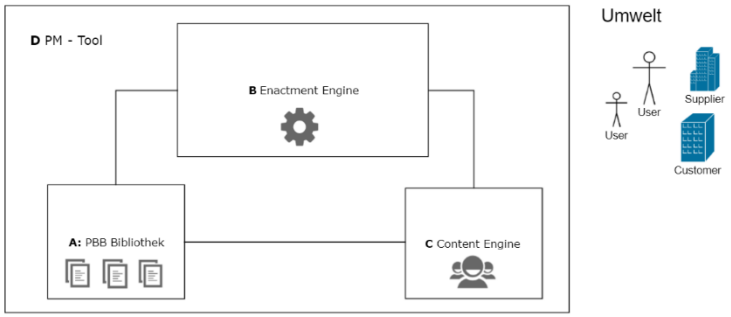


Figure 7 PM-Tool system structure (own representation)

System D - the PM-Tool itself consists of three core components:

- PBB (Process Building Blocks) library
- enactment engine
- content engine

The core components are to be regarded as separate subsystems. Their structure and properties are specified in the following subchapters.

2.2.1 PBB - Process Building Blocks

The PBB library contains the so-called PBBs (process blocks). In the project context, a PBB can also be described in a simplified way as a work package or as the (work) ability / capability of an employee. In the context of the process management tool, these are modules from which process chains are formed when stimuli are received. They represent the capability and the action concepts of the project resources. Since the rollout of the PM-Tool, the library already contains more than 13,000 different process building blocks.

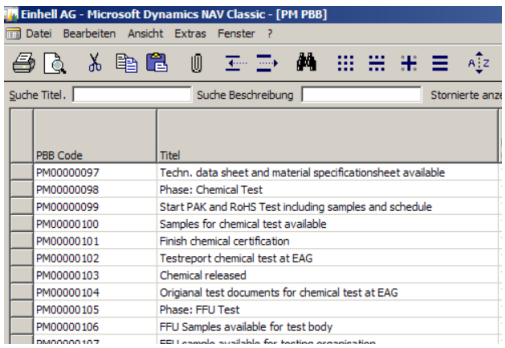


Figure 8 PBB library (screenshot PM-Tool)

A domain structure has been implemented in order to be able to delimit the content of the PBBs. This resulted in sub-libraries for various specialist areas such as quality, product management or

engineering. In the overall library, the individual modules are differentiated by an independent numerical code. The allocation is automated when a new PBB is created.

A single PBB is in turn composed of four core elements. The macro level (A1), the micro level (A2), the content environment (A3) and the wrapping information (A4).

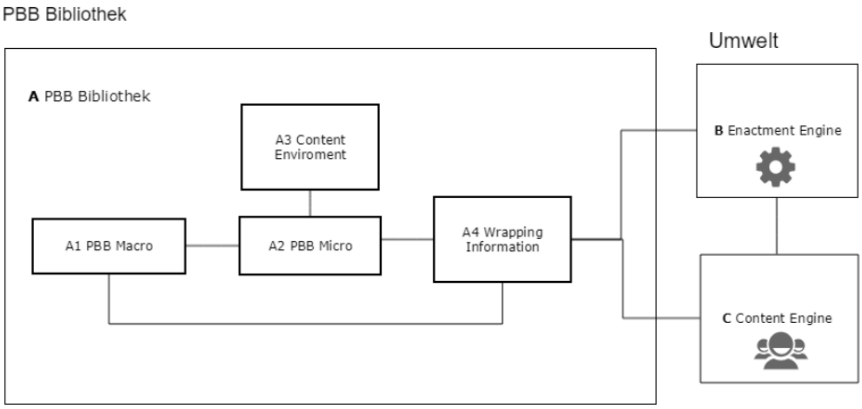


Figure 9 PBB library system structure (own representation)

PBB Macro (A1)

The naming of a work package takes place at the macro level (A1) of the PBB. The naming depicts the sum of the things to happen at the micro level.

PBB Micro (A2)

The micro level (A2), on the other hand, answers the question of how the work package is to be carried out in detail and what is needed to fulfil it. Currently, the system can map three different possibilities at the micro level. The simplest form is a Standard Operating Procedure (SOP). SOPs are descriptions of a work step (work instruction) in text form, as they are also used, for example, in flight safety instructions (Scheiderer and Ebermann, 2010, p. 25). A further characteristic at the micro level is the addition of further information content to the PBBs in the form of key value pairs (KVPs). The keys are pre-implemented and the values are filled later by the processor of the PBBs. The KVPs represent a language construct, which spans up a content level.

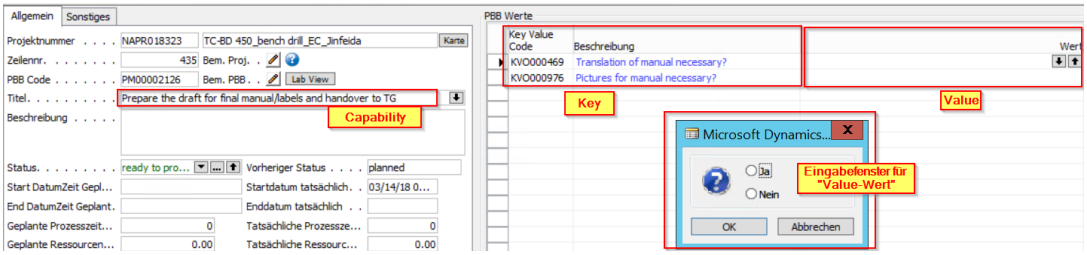


Figure 10 PBB macro level and micro level (screenshot PM-Tool 19.05.2018)

Last but not least, a classic checklist can be embedded in a PBB to ensure that no important component is missed to fulfil the task. The example below shows a PBB where the resource must request and save certain documents from the supplier. Using the checklist in the PBB table, the agent can ensure to check anything that is required to ensure that the document transfer is complete. By clicking "Yes", he can confirm and track the receipt of the individual documents.

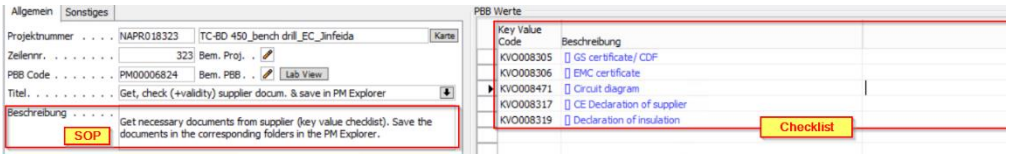


Figure 11 Micro level: SOP and Checklist (Screenshot PM-Tool 19.05.2018)

Closely linked to component A2, the Content Environment acts as the third element. The content environment defines the sources from which data can be pulled into the PBBs. In other words, it represents the interface to the content level. Currently, the data from the Product Data Management System (PDMS) of the EAG can be reflected into it. Furthermore, it is also possible to display values from other PBB keys as so-called Result KVPs. If certain values of a preceding module are relevant for the subsequent module, this is reflected for the processor as result KVP. However, this only applies to PBBs within the same project.

Wrapping Information (A4)

The PBBs are equipped with a set of additional information - Wrapping Information (A4) The wrapping information acts as a boundary element and enables the enactment engine to select the correct PBB at a given point in the process. In today's PM-Tool this is solved by assigning links to other PBBs. Furthermore, the wrapping information also includes the function of attaching conditions to the individual PBBs. In conjunction with the template item, parallel processing, sequencing, and the sequence of the PBBs can then be realized.

2.2.2 Enactment Engine

The engine of the PM-Tool is the Enactment Engine. It functions as a real-time system and processes all the requirements that are placed on the system. To get a new project rolling, a system interaction with the user is necessary. The user specifies concrete parameters such as the project types or product code within a ready-made project mask. The Enactment Engine processes this information in the background and combines the individual PBBs into a process chain. The wrapping information supports this process and enables the Enactment Engine to consider pre-assembled PBBs.

Now it is not only crucial to be able to construct a seamless process, efficient resource management is also essential.

For this purpose, a common understanding of the implemented role model must first be created. Individual roles have been named for all departments that work with the PM-Tool throughout the Group. The designation was based on the users' respective areas of responsibility. The parent company alone therefore lists more than 100 different roles within the PM-Tool. In a second step, the users were assigned to the roles. The example below shows the role of the product managers - in short form called "PM-EAG-PM". In the column "Assigned users / role / group" it can be seen that 21 users (= 21 product managers) are assigned to this role.

Benutzer/Rolle/Gruppe	Name	Type	Datenbank	Status	Zugeordnete Benutzer/Rolle/Gruppe
PM-EAG-PM	**Product Manager EAG**	Rolle	EAG	aktiviert	21
PM-EAG-SEC	**Product Manager EAG Secretary**	Rolle	EAG	aktiviert	1

Figure 12 Examples of roles from product management (screenshot PM-Tool 21.05.2018)

In the role "PM-EAG-SEC" again a user. The roles are already hard-wired to a PBB (Figure 13), the users in turn are wired with product codes (Figure 14).

B Code	Baum	Baum alle	Titel	Beschreibung	Pflege durch Dat...	Verantwortlicher	Verantwortlich Ro
400006210			| | | | | Review and complete all Data in PDMS		EAG	ERIC.PAGE	TP-EAG-PE
400006211			| | | | | Review and complete all Features in PIM		EAG	MICHAEL.GITSCHEL	PM-EAG-PM
400006392			| | | | | Provide Cumulus Assets to IPM	use excel sheet template	EAG	RAMONA.FINK	PM-EAG-SEC
400007385			| | | | | Check Cumulus Excel-Sheet and define Cumulus ranking		EAG	MICHAEL.GITSCHEL	PM-EAG-PM
400006212	▾		| | | | | Start packaging briefing		EAG	MICHAEL.GITSCHEL	PM-EAG-PM

Figure 13 Role and PBB relationship (screenshot PM-Tool 21.05.2018)

Tabellen Link Code	Code	Beschreibung	Rolle	Benutzer	PM User Name
ITEM SU...	B1150	Power Generator	BUM-CM-EAG	GUENTER.SCHIEWIETZ	Günter Schiewietz
ITEM SU...	B1160	Table Saw	BUM-CM-EAG	GUENTER.SCHIEWIETZ	Günter Schiewietz
ITEM SU...	B1170	Transportation Tools	BUM-CM-EAG	GUENTER.SCHIEWIETZ	Günter Schiewietz
ITEM SU...	B1180	Work Shop Equipment	BUM-CM-EAG	GUENTER.SCHIEWIETZ	Günter Schiewietz
ITEM SU...	E2010	Cordless Drill (not PXC)	BUM-CM-EAG	SABINE.BUSLER	Sabine Busler
ITEM SU...	E2020	Cordless Multifunctional Tool	BUM-CM-EAG	SABINE.BUSLER	Sabine Busler
ITEM SU...	E2030	Cordless Impact Drill	BUM-CM-EAG	SABINE.BUSLER	Sabine Busler
ITEM SU...	E2040	Cordless Screwdriver	BUM-CM-EAG	SABINE.BUSLER	Sabine Busler
ITEM SU...	E2042	Drywall Screwdriver	BUM-CM-EAG	SABINE.BUSLER	Sabine Busler

Figure 14 Example of users assigned to product codes (screenshot PM-Tool 21.05.2018)

The product code must be filled in as a mandatory field in the mask before the project starts. Thereupon the explicit user selection can be made by the Enactment Engine.

If a project resource fails or is absent, this is currently being resolved by a substitution arrangement. Each user can store his or her substitution in the system and activate it if required.

The Enactment Engine maintains a "to-Do List" for each user. It guides the employees through the working day and manages the sequencing of the work of each resource.

As the last point in the area of resource management, the engine performs a measurement of the used resource time in the background. The measurement takes place while the user actively calls the process building block in the system for processing. The measurement result is then proposed to the user. The user can accept the proposed time or make corrections, since the engine cannot take exogenous factors such as incoming telephone calls into account.

At the same time, the Enactment Engine measures the process time, i.e. the time consumed by each task.

By establishing a project visualization level, the Enactment Engine supports the project manager in all project monitoring and controlling tasks. This query view is called "Matrix View" in the PM-Tool and provides a tabular overview of all projects. A great advantage is that the user can configure the view individually according to his needs.

The to-do lists are provided for all users in a visual form. In this list, the blocks that are due for processing are first displayed on the macro level (see Figure 15).

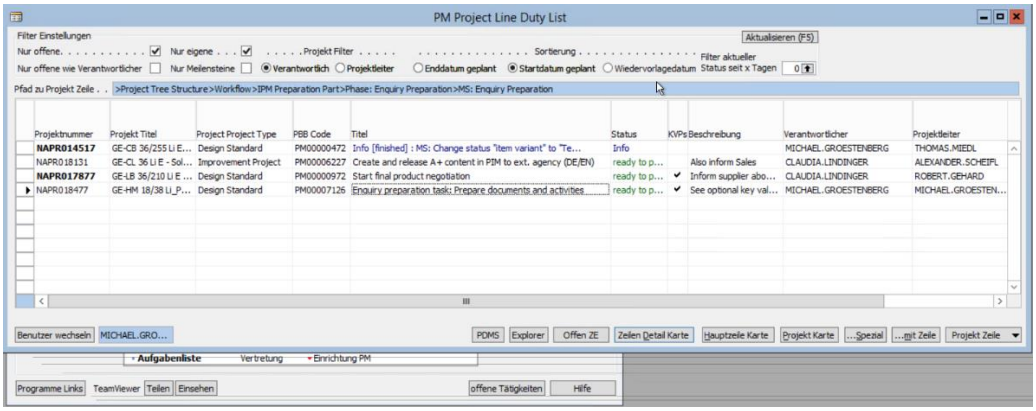


Figure 15 "Duty List" task list (screenshot PM-Tool 22.05.2018)

To get to the micro level, the PBB must be opened by the user in a separate step.

The Enactment Engine imports all PBBs relevant for processing into the task list. In this area, the user can manage the tasks independently and perform resource time booking.

2.2.3 Content Engine

The Content Engine is the third core element of the PM-Tool, whose supporting function is documentation. It can be understood as a kind of "room" or "warehouse", in that all the contents of the projects are stored that arise during the processing of the PBBs along the process. Those

fingerprints that are left behind during the run of processes are stored by the Content Engine in a form accessible to the PM-Tool user. For example, entered KVP values can be displayed.

Due to the embedding in the ERP system Navision, the language model, which represents the input for the content engine, is kept relatively simple. It is a classic key-value system with a predefined number of columns and associated values. Different views from the content "warehouse" of the project can be individually compiled in a matrix view.

2.3 Sample process from the department Supply Chain Integration

The example process is used to explain the procedure for introducing a new process in a dedicated department in more detail. This also illustrates how the tool is embedded in everyday business.

2.3.1 Professional background and motivation for using the PM-Tool

The Supply Chain Integration (SCI) department made use of the PM tool at the beginning of the year. This was triggered by a project involving a group-wide rollout of the "Diskover" material planning software. The project includes IT tasks such as the daughter-specific adaptation of the software following a fit-gap analysis. In the same way, technical actions are also part of the rollout. An example of this would be the end user training of Diskover.

Up to now, rollouts have been handled in a classic project organization, without a PM tool. However, the general progress of the project was slow. Major project weaknesses are waiting times, recurring errors or a lack of transparency of the departments during the entire implementation (Zollner, 06.07.2018).

Motivated by the fact that the “Strategy 2022” initiative provides for a digitalized process organization, the department started implementing the rollout project of the PM-Tool in January 2018. The SCI department saw this as an opportunity to reduce the previous difficulties during the project process.

2.3.2 Steps towards implementation

The introduction of the PM-Tool into the SCI area was carried out in four steps:

1. Presentation PM-Tool (and its mode of operation)
2. Formulation of the PBBs (Templating)
3. Transfer to the PM-Tool
4. Detailed training of the users

Due to the fact that the related staff involved had no experience with PM-Tool, a general introduction of the PM-Tool was first carried out. Although not in the conceptual depth as described in chapter 4.2. The goal of the one-hour presentation was to explain the advantages of using the tool and thus to increase the acceptance of the tool.

Subsequently, the individual PBBs were defined and specified in a separate meeting (see chapter 3.4). The most proven method here is a mixture of open discussion and workshop with the project participants. In addition, the already existing project plan was used as a basis for discussion, which was visually presented in a flow chart. Gradually, the individual PBBs were specified in more and more detail, even at micro level, and grouped into a total of eight milestones. Due to the lack of experience of some of the participants, the effort for this undertaking took about three working days. The result was a workflow prepared in Excel consisting of 163 PBBs including all necessary meta-information, such as the chronological processing sequence and seven new PBB roles.

The Excel version of the workflow was subsequently passed on to the process manager, whose responsibility was the technical implementation in the PM-Tool. The effort required for this was calculated depending on the queries that arose and the length of the entire workflow. As a rule, two to three working days are expected. In the case of the Rollout Template, however, the effort required was five working days. Reasons for this were, among others, the necessity of a new, separate project group and new roles. Likewise, there were no reusable work packages in the PBB Library that were suitable for the project.

The last step was a user training of the tool. This explained in particular the use of the task list and project status tracking using the Matrix View. As an example, the new, already implemented workflow was used to provide the possibility of a plausibility check by all participants at the same time.

2.4 Strengths and weaknesses

In the following chapter, the strengths and weaknesses of the current PM-Tool are explained. These will also be taken into account in the later development of a user spec for a new PM-Tool. The sources of the mentioned strengths and weaknesses are based on a survey

2.4.1 Strengths

The possibility to standardize business processes in the departments by using predefined templates and defined PBBs is one of the main strengths of the PM-Tool - combined with the fact that the project progress is automated or semi-automated. According to an internal evaluation, the coordination effort between the departments can be reduced by up to 30%. By reducing administrative activities, more time is available for value-adding work or analytical activities that contribute to the company's further development.

With almost 1200 projects handled annually in technical product development alone, the automated scheduling of tasks is a considerable advantage in addition to project continuity. The current PM-Tool can thus ensure that the upcoming task packages arrive at the right time at the right resource.

In the area of project monitoring, the task list is considered a great support in everyday work, as well as the matrix view as a means of monitoring all projects.

Another strength is the centralization of project documents. The predefined folder structure, which is linked to each running project, can be accessed by all users of the PM-Tool throughout the group. This makes it easier to exchange data such as important supplier e-mails, contracts or product certifications. An advantage is the reduction of time-consuming e-mail communication. The figure shows a screenshot of a folder structure from the technical area.

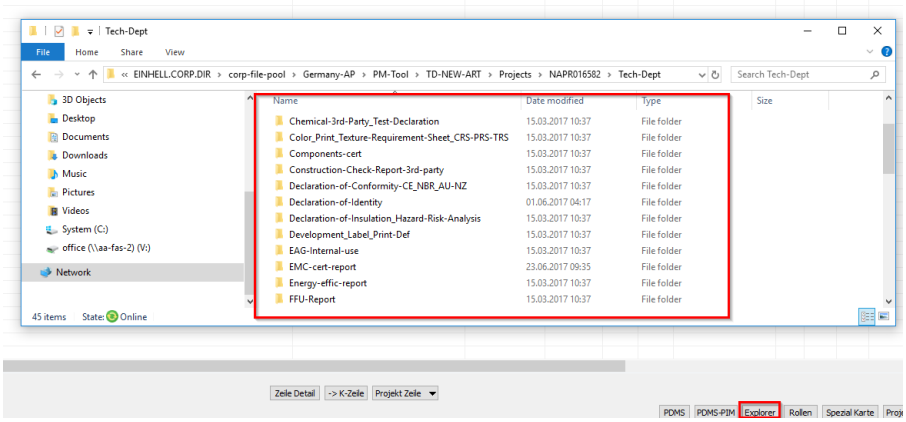


Figure 16 Centralized data exchange of relevant project data (PM-Tool Screenshot 13.08.2018)

2.4.2 Weaknesses

After the initial introduction of the PM tool in the department Supply Chain Integration (see chapter 4.3), one encounters some limitations of the tool. One of the main reasons for this is the fact that the tool was very much tailored to technical product development (e.g. cordless screwdrivers, lawnmowers) and engineering. What was good in the past (PDMS as Domain Data Model) is now disturbing for the introduction of the tool into new departments (no possibility to implement new Domain Models or related role models, many fixed header data fields dedicated to engineering, etc). This can be seen, among other things, in the so-called project header, where many fields have been predefined to product requirements. The figure below shows, for example, the "Info" tab, which will remain largely empty during a rollout project, since the corresponding data does not exist in this context.

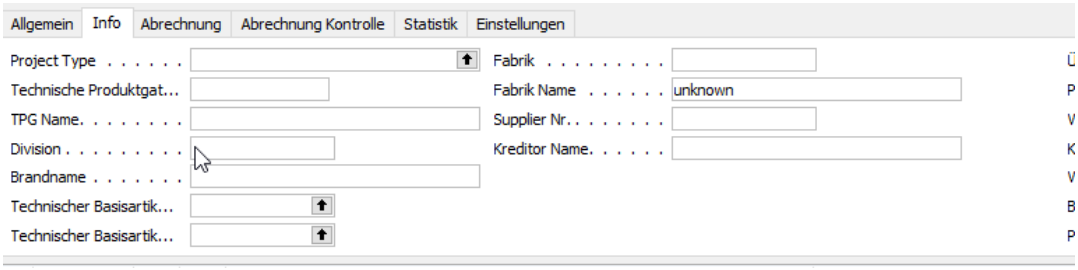


Figure 17 Product-oriented data fields PM Tool (Screenshot 14.08.18)

There are also limitations of the current PM-Tool in terms of functionality. For example, only a forward calculation can be carried out automatically for project scheduling.

Another problem is the separate databases and the different types of access to the PM-Tool. The difficulty for the users here is that synchronization problems between the databases occur more often and therefore the project status might at times not corresponds to reality correctly. For example, a user in China has already set his task to the status "finished", while in Germany it is still displayed as "in progress" until the next synchronization takes place.

The different types of access result from the fact that the PM-Tool has been embedded in the ERP system. However, the Group currently holds different versions of the ERP system from subsidiary to subsidiary. As a result, the user interfaces of the PM-Tool vary and make it difficult to coordinate employees and cross-company training of processes.

However, the biggest improvement points are in the area of tool performance, usability and design. Users gave negative feedback on this and rated the tool as slow, complicated and confusing. The fact is that the loading times are subjectively considered too long and the menu navigation is little or not at all intuitive.

A missing search function leads to the fact that due to the high information content of the tool, the user loses a lot of time to find certain KVP values from older projects.

The design is generally outdated. A web-based or even mobile version with the current means and environment is unthinkable based on the given technology stack.

2.5 Selected PM Database Statistics – As of May 2020

Date: 8. Mai 2020

Active users: 579

- ***Project Group: EC-QA-LABT QA - Labtest EC***
  - **Number of projects: 25.438**
  - **Number of lines complete: 787.438**
  - **Number of lines with result KVPs: 430.828**
  - **Number of result KVPs: 3078.291**
  - **Number of test samples: 48.313**

- ***Project Group: TD-NEW-AR TP - New Article Project***
  - **Number of projects: 15.959**
  - **Number of lines complete: 2.145.063**
  - **Number of lines w/o classification: 1.482.615**

3 Conception of the new process management tool

The following figure shows a phase model for the concept development for the new edition of the process management tool, which was jointly developed.

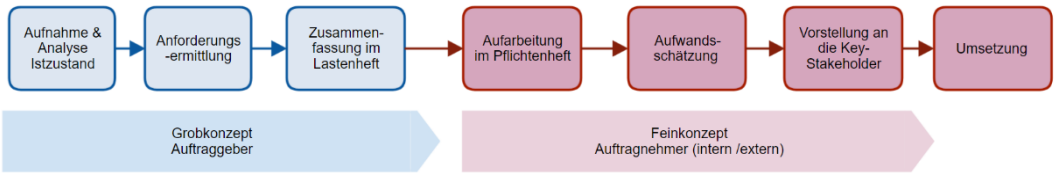


Figure 18 Project management procedure (own presentation)

3.1 Recording and analysis of the actual state

In the first phase, the current state of the system is recorded and analyzed. The aim is to define the system context for which the specifications are to be created in a later step. It is crucial to record the most important core elements of the system as well as the structural conditioning of the components. The relationships to the environment, such as existing interfaces to other systems, must also be considered. For this purpose, expert discussions were held with the developers and users of the existing PM-Tool in the PM-Tool project.

3.2 Requirement determination

Phase two deals intensively with the task of requirement engineering in order to achieve a qualitative improvement of the system. For this purpose, the relevant stakeholders are determined in the first step. The goal is to determine all influencing variables that are directly or indirectly affected by the project.

In the case of the new PM-Tool concept, this includes the following organizational units: Engineering, Category Management, Corporate Purchasing, Marketing, Quality Assurance and Supply Chain Integration. From the group's point of view, the ISC GmbH with the department Service International and IT, as well as the spare parts department are also to be seen. Furthermore, the subsidiaries Hans Einhell (Shanghai), kwb Germany GmbH (Stuhr) and Ozito Industries Pty. Ltd. (Melbourne) are also involved in the project.

Due to the larger number of stakeholders, a combination of different methods was chosen to determine the requirements. On the one hand, an online survey with a total of seven questions was designed, two of them closed and five open questions. With the help of the survey, it is intended to generate the highest possible coverage of all previous end users of the tool.

The predominantly open design of the questions ensures that both critical feedback and suggestions for the tool can find space and be formulated. For example, one of the seven questions is: "Three functions / approaches that should be retained from the current PM tool? (SurveyMonkey Europe UC).

During the evaluation, frequently mentioned words / expressions are grouped and analysed. This makes it possible to identify the points where users subjectively see the greatest added value in using the tool. These points are then incorporated into the specifications as requirements. The survey was online for exactly seven working days.

An additional input from a total of 41 participants was generated. The completion rate was 100% and the average processing time was twelve minutes.

On the one hand, separate interviews were conducted with selected experts on the basis of a previously developed guideline. This should provide a deeper insight into the requirements for the new PM tool. The experts cover a small sample of end users and IT.

3.3 Combination in the specifications

Once the requirements had been collected, the next step was to analyze, sort and summarize them. The aim of the specification sheet is to specify the necessary requirements so that the client (external or internal) can develop solutions with as few queries as possible.

The specifications are divided into five blocks:

- Basic information
- Status Quo: Application areas of the current PM tool
- User groups
- Interfaces
- Requirement description

It begins with basic information and key data such as the purpose and objective of the specifications, budget limits or even contact persons in the company. Furthermore, existing areas of application of the existing tool are outlined and the stakeholders are listed.

The selection of suitable system boundaries, but also the relationships to the system environment, is a decisive factor for a successfully functioning system. Therefore, the topic area of interfaces was concretized in a separate chapter. This should enable future technical developers to consider the potentially necessary interfaces (system-environment relationships) in detail when working out the requirements specification.

The core of the specifications, the functional and non-functional requirements, are dealt with in the last block. The following two sub-chapters present an excerpt of the requirements, which have resulted from the phase of requirements determination.

3.3.1 Functional requirements

The functional requirements describe what the system has to do or which external interactions, for example by a user, should be enabled.

Process planning

One of the key requirements is the expansion of the functionality in the area of automated time planning/Scheduling. In addition to the previous forward calculation, the new tool should also be able to calculate projects backwards. The need arose repeatedly from product development, but also from the SCI rollout project. More and more often, projects have to be completed on fixed deadlines. In product development, for example, this applies to seasonal products (gardening season), whereas for rollouts, predefined "desired" go-live dates are relevant.

Based on this, the individual PBBs could be equipped with granular times, such as the minimum, maximum and average processing time (process time-based). The origin of this requirement is mainly in the area of sales and distribution. The minimal processing time allows the project planner more flexibility to carry out shorter calculations for projects on which the focus has been placed, and thus to meet current market conditions.

Process Execution

A so-called "Cross Projecting" should be the basis for a more intelligent project management. Cross Projecting should enable projects to communicate with each other and to interweave process threads. The advantage of cross-projecting is, for example, in rollout projects such as those currently being carried out by the SCI department. In the case of rollouts at subsidiaries, it can happen that features specified by the country are relevant for the entire group. Therefore, a general update in the core system is necessary. However, the processes for product management for Diskover would possibly run in a different domain. For both running processes, however, the view on the implementation of the required feature is important.

Furthermore, not only automated PBBs are to be triggered, but also sub-processes. From a purchasing perspective, for example, the process part has already been completed; from a technical perspective, the decisive sub-area is just beginning.

For learning the system, it is also important that the user can give ratings for the whole process or for individual PBBs. Based on this, the system should suggest optimization potential later on.

Process monitoring

Process monitoring is a central issue in a wide variety of corporate divisions. Therefore, monitoring must be made possible similar to the previous Matrix View. A "Project Dashboard" should provide better visualization and be able to display diagrams, for example. The project dashboard should also include an individual design by the user. The user can then display the required process key figures in various forms.

In addition, a reminder function generated by the system should relieve the project management of the mails they are currently sending out if a PBB deadline has been exceeded. The reminder is to be sent automatically to the person responsible for the PBBs and is to indicate that the work is due soon or already due.

Furthermore, the system should also be able to determine the critical path for each project in order to enable improved and more targeted project monitoring.

Process documentation

The Content Engine is the documentation level of the current PM-Tool. The element is to be extended and will be able to display a central change history across all processes.

The new tool must also be able to record resource times. A system-side time recording along the process enables the costs of the PBBs to be charged to internal or external sources. Furthermore, process-oriented key figures such as productivity or efficiency are to be measured in order to be able to draw conclusions about the performance of departments.

The key figures determined by the system, such as the above examples from the specifications, should be able to be validated by users and adjusted if necessary.

A central search function is intended to support users in various searches for past and significant information content.

3.3.2 Non-functional requirements

The non-functional requirements are specifically concerned with the question of how the system should implement the technical expectations so that a satisfactory applicability for the users is given. The applicability is one of the main prerequisites for the later acceptance of the system.

Quality

The non-functional requirements start in the specification sheet with the expectations of the system quality. Among other things, the quality of the system is largely determined by performance and efficiency. With regard to these specifications, an identical standard must be created throughout the

Group. The primary requirements in this area are shorter response times and 100 percent availability of the future tool. It is therefore imperative that a concrete proposal for a solution to the decentralized database problem can be drawn up.

Ergonomics

During the requirements assessment it was also determined that the introduction of a common language would simplify communication across companies. English is to be claimed as the leading system language - with the option of being able to display translations in Chinese or German and thus still allow multilingualism.

A further point for improving the ergonomics is that the new tool is equipped with modern Graphical User Interfaces (GUIs). Examples are functions such as "drag and drop", "scroll" or even "zoom". There is also a desire for real-time communication, similar to an online chat.

Introduction, use and support

A necessary boundary condition for the further development of the tool arises during the introduction, use and support. Thus, a solution concept is to be worked out how already running projects can be transferred from the current PM-Tool to the new PM-Tool.

During the new development, it should also be noted that a corresponding user manual will be prepared which will be accessible online. Parallel to this, a software documentation must also be kept in which all technical concepts and further developments are anchored.

Although the new system requires a high degree of intuitive operability, a helpdesk should be available for possible questions or system problems.

Modifiability

For future developments and changes to the system, an architecture is required that consists of independent modules so that individual modules can be added or removed at any time.

PART III: User Story - “Functional Description From A User Perspective”

1 Basic information

The specification sheet explains the objectives, tasks and key data of the project. The foundation is formed by the documentation of the actual state with subsequent explanation of the target state. The specifications define which problems require concrete solutions. The technical result document is to be seen as a specification sheet for the further implementation phase of the contractor (see figure 1).

Figure 1: Approach project plan

1.1 Benefits and objectives

The further development of the digital process organization in the company in relation to the changing conditions of the company organization and its environment. Any organizational unit with methodically recurring contents in its field of activity should be able to use a process management tool efficiently.

The basis for this is the existing process management tool concept, which has been in use for over ten years.

The main objectives include:

- Massive increase in user-friendliness
- Keyword: Intuitive software guidance
- Changeover to a modern software design (Graphical User Interfaces, web-based Design, Live Chats, Dashboards, App)
- Retention of conceptual strengths
- Addition of missing functions
- More flexibility and simplicity in process design and execution, e.g. through own process design by the specialist department, while maintaining a standard

1.2 Project participants and key data

1.2.1 Deadlines and cost framework

The project was defined as a target implementation by the end of 2021. The cost framework will be specified in the contracting phase.

1.2.2 Project participants & contact persons

Name	Function	Contact
Dr. Markus Thannhuber	Chief Technical Officer	Markus.Thannhuber@einhell.com
Christian Neu	Senior Process Manager	Christian.Neu@einhell.com
Christoph Kufner	Software Developer	Christoph.Kufner@isc-gmbh.info

2 Status quo

2.1 Application areas of the current PM-Tool

- Product development and preparation process
- Laboratory tests
- System rollouts
- Pre-Shipment Inspection (PSI)
- Improvement projects
- Claim management processes
- Mass Production Quality Control (MPQC) processes

2.2 Companies

- kwb Germany GmbH
- iSC GmbH
- Hans Einhell (Shanghai)
- Ozito Industries Pty Ltd.

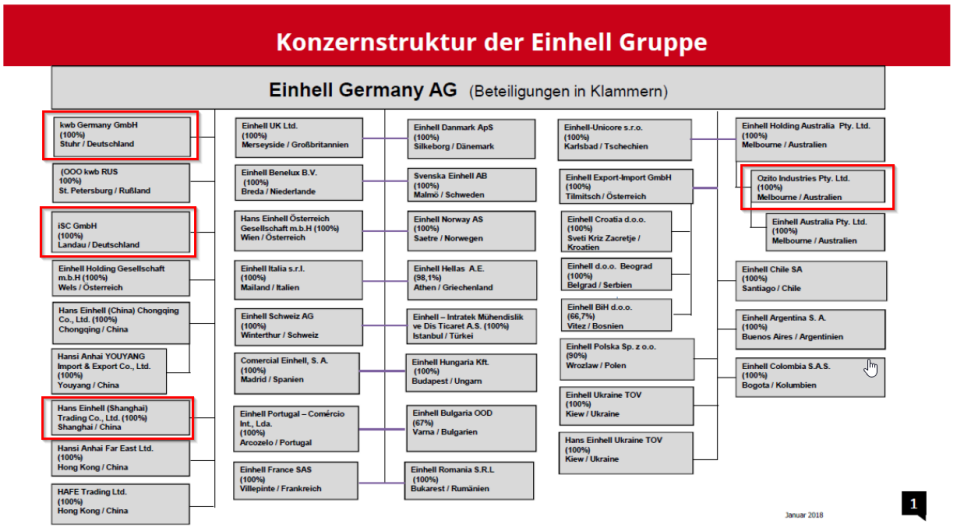


Figure 2 Group structure of the Einhell Group

2.1.1 System structure

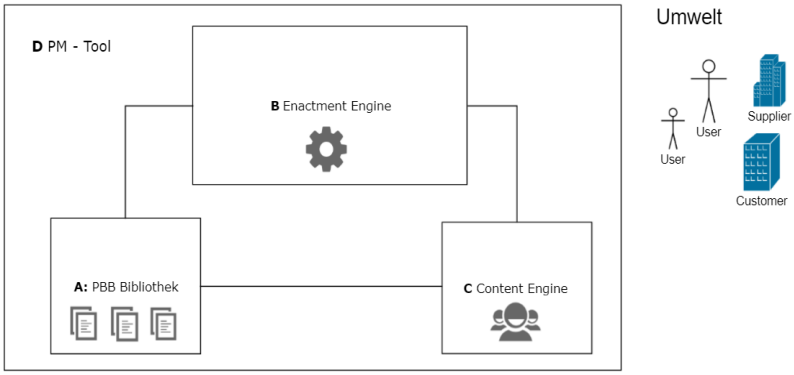


Figure 3 Main elements of PM-Tool

Overview: Main elements

	Elements	Description
A	PBB Library	Library of the process blocks
B	Enactment engine	Executing level
C	Content engine	Documenting level

Structure of PBB - Library

Elements	Description
Macro level (part I)	Work package (What should be done?) and process integration
Micro level	Execution of the work package (How […]?)
Wrapping Information (Macro level part II)	Metadata (Who and when?) –> Linking Ensures that the process integration works on both the output and input side

3 User groups

3.1.1 Internal

The current focus was on the parent company, as the product ranges were planned and prepared centrally there. As the system is to move away from the strong product focus or is to find a broader field of application, the target group (internally) is to be considered as group-wide, with all departments having to manage the processes. Likewise, the core process of product development and engineering should be possible decentrally.

3.1.2 External

The active involvement of users outside the Group, such as customers, suppliers, advertising agencies or partners, should be made possible.

4 Interfaces

Networkability with other applications:

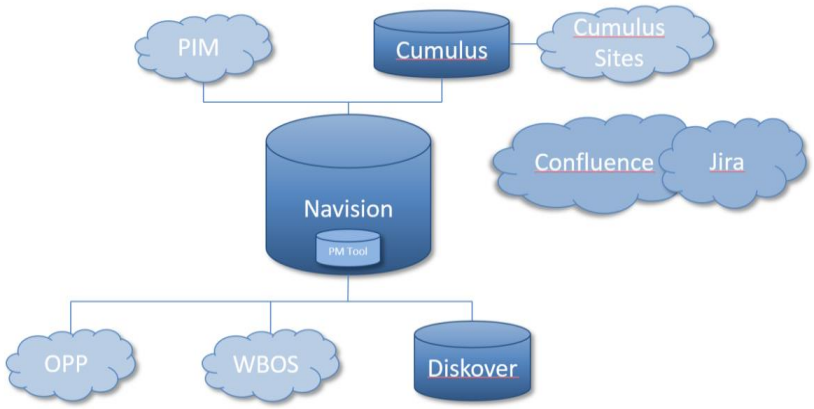


Figure 4 Extract from the EAG system world

Confluence	*online wiki*
Cumulus	*digital asset management*
Discover	*Material Planning / Dispatching Software*
Jira	*Agile project management*
Navision	*ERP system*
OPP	*online purchase platform*
PIM	*product information system*
WBOS	*web-based ordering system*

In the course of preparing the specifications, it is to be examined whether the new PM tool could replace existing applications (such as OPP) in their entirety.

In addition, the possibility of linking with MS Outlook, Saperion (archiving system) and Bedatime, which is a resource time management system of the EAG, is to be examined.

5 Requirement description

5.1 Functional requirements

5.1.1 System structure (TARGET)

5.1.1.1 Domain structure / Security structure

The process modules should be delimitable in terms of content using a domain structure.

5.1.1.2 Introduction of a process module Responsible person

Each process block should have an owner and a co-owner. The owners are later responsible for the topicality and for the contents of the PBBs. In a regular review cycle, the topicality must be checked, guaranteed and adjusted if necessary.

5.1.1.3 Central Capability Management

A central capability management is to be anchored in the new system. This means that a training table will be maintained for the process modules, in which the following is listed: Who was trained for which module? When, how often and how long? Not only the training aspect, but also how often the module was processed by whom and in which quality must be considered. By means of both key figures (training + number of projects) an "overall score" should be determined. As a result, the number of "experts" or "newcomers" can be monitored and improved resource management ensured.

5.1.1.4 Modularity of the PBB micro level

For the execution of a PBB, it should be possible to hang in several options for processing, depending on the requirements of the organizational unit.

Examples:

- Checklists
- Micro-workflows (sub-process in PBB)
- Standard descriptions (Standard Operating Procedure)
- SCRUM development cycles
- Forms
- Access to other interfaces
- Direct connection of expert systems
- Office Robot Applications

Depending on the degree of complexity and the socio-cultural framework, it should be possible to define freedoms more narrowly or more broadly

5.1.1.5 Execution of work modules within the PM-Tool

All execution steps that are necessary for the execution of a process should take place within the PM tool. Both the input and the value creation should be able to take place within the system boundaries of the PM tool (e.g. creation of a catalogue). This is to avoid a constant change of systems. The processing should be possible in the module e.g. sending of e-mails or the archiving of files in Saperion. Consequently, the internal binding of the system elements is strengthened and links to the system environment are kept simple.

5.1.1.6 Wrapping Information: Designing more flexible role assignment

The wrapping information, which wraps the PBB with information, specifies, among other things, the role responsible for executing the process module.

	Projekt Gruppe	PG Beschreibung	Benutzer/Rolle	PM User Name	Berechtigungs Art	Status	Rollen Benutzer Zuweisung durch Rolle	Zuweisung während Projekt
	TD-NEW-ART	TP - New Article Project	**TP-EC-AU-IP**	**EC Ozito Improvement Project**	Assistent	aktiviert		✔
	TD-NEW-ART	TP - New Article Project	**TP-EC-AU-PC**	**TP-EC-AU Project Coordinator**	Projektleiter	aktiviert		
	TD-NEW-ART	TP - New Article Project	**TP-EC-AU-PE**	**TP-EC-AU Project Engineer**	Projektleiter	aktiviert		
	TD-NEW-ART	TP - New Article Project	**TP-EC-AU-PMO**	**TP-EC Australia PMO**	Projektleiter	aktiviert		
	TD-NEW-ART	TP - New Article Project	**TP-EC-CE**	**TP-EC Certification Engineer**	Assistent	aktiviert		
	TD-NEW-ART	TP - New Article Project	**TP-EC-CN-TEST**	**Test role for Christian Neu**	Assistent	aktiviert		
	TD-NEW-ART	TP - New Article Project	**TP-EC-DISC-PMO**	**TP-EC Discounter PMO**	Projektleiter	aktiviert		
	TD-NEW-ART	TP - New Article Project	**TP-EC-DM**	**TP-EC Division Manager**	Assistent	aktiviert		

Figure 5 Section from the current roll table

Users are also stored behind each role. Technical product class codes, for example, are stored behind these users. When starting a project for a cordless screwdriver, for example, the correct user can be assigned to a task (wrapping information).

However, flexibility is limited and should be able to process several combinations of meta-information, e.g.:

- If cordless screwdriver + customer X, then user A and if cordless screwdriver and not customer X then user B
- It should be possible to assign several different roles to a task in the template, e.g. if it is a New Trade Article then "Project Coordinator", otherwise "Project Engineer".
- Role assignment only during the course of the project e.g. supplier is only determined in the course of the project
- Generally different modelling possibilities for other domain-oriented domains

5.1.1.7 Different possibilities of PBB assignment; dynamic assembly and learning processes

In the current PM tool, PBBs are cained up by the use of fixed and conditional links leading to process sceletons that again can be used as templates. PBBs here take positions in the skeleton or likewise the template but might be governed by logical conditions. This point is to be extended and offer several possibilities, which are described in the following subchapters.

5.1.1.7.1 Learning system: Statistical procedure (AI-supported)

A statistical procedure was discussed in the IT workshop. The considerations were based on first modelling experiments in the laboratory area in Hong Kong. There, a statistical method was installed, which allows to extend the PBB selection possibilities during the process flow. The PBBs were modelled three-dimensionally. The PBB has an entry point and several possible exit points.

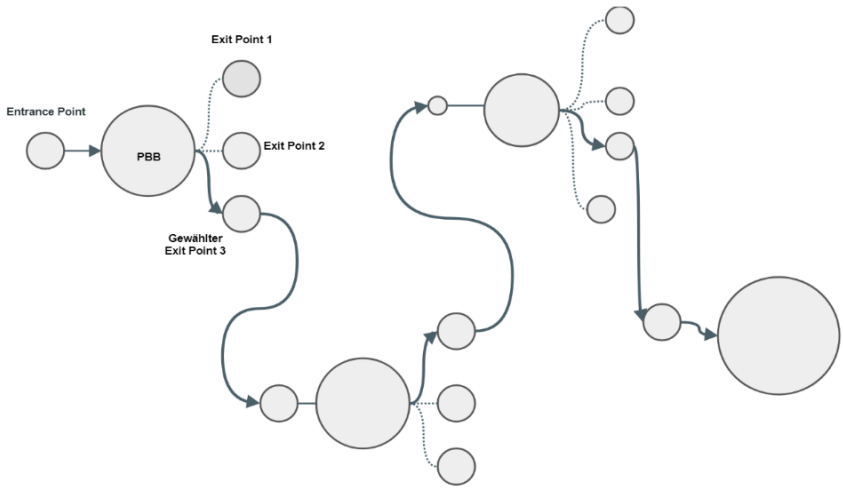


Figure 6 Learning process chains

The exit points represent potential follow-up modules. Furthermore, each PBB was assigned to a “physical mass”. With increasing age or frequency of use, the mass was increased. Exit points were assigned a mass, too, but weighed less. After project completion, a “physical impulse” was injected, with which involved (chained) entry and exit points of PBBs moved towards each other (good project result/score) or separated (bad project result/score) depending on their mass. Gradually, the distances of the respective points from each other in space are thus shortened for successful chaines and finally a preferred process chain is established.

5.1.1.7.2 Proposal lists

PBBs are available, the user has to decide. Especially for new processes that are not yet established, decision paths at a higher level should be enabled.

5.1.1.7.3 Hybrid form

A mixed form means to combine the logic-driven approach and the statistical approach. For example, in the course of system rollouts, there are corresponding training courses which, according to the logic, can only be carried out after successfully installed test systems. At the same time, the PBB library contains several blocks that reflect the work content of a software training course. However, these are different at the micro level. Here, statistics could support the training by:

- pre-selecting the training modules

- rating according to relevance
- providing pre-filtered lists to end-users

In the next step, the end user can choose the most appropriate PBB for his case.

5.1.1.7.4 Enactment Control: not only parallel / serial

When placing PBB links, compromises often have to be made between chronological order and logical coherence. For this reason, certain activities may have to be triggered manually in the process flow, since their content is anchored at a different point, but could start earlier.

An additional control option should be made available. Currently there is a limitation by levels. The new level only starts when the entire previous one has been processed. This limitation must be replaced by a more flexible variant.

Example:

When PBB X gets the "finish" status, PBB Y (which is in a completely different location) starts immediately as well, because it can already be edited here, even though its content belongs to a different milestone. -> Event based Gate Triggers!

5.1.1.7.5 Dynamic assembly

The processes are to run so intelligently that the system can draw the required building blocks from different domains by itself. Each process has its own characteristics and even if they usually have certain similarities, each one has its own individual requirements.

The person who initiates the process should be able to design the required process by means of a kind of construction kit. Currently, this is represented by CIPs, which, depending on the value specified, influence the further course of the process and set milestones, for example, to "not necessary".

5.1.1.8 Cross Projecting und Project Consolidation

Cross-projecting is a requirement that arises from the process of technical product development, among other things. The maintenance of an activity data model should enable a link between related projects.

For example, a power tool is being developed that is to be exported to various countries. One of the countries is the United Kingdom. Here, a different type of plug is required and, as a result, different test procedures than in Europe. In today's PM-Tool, this case can only be mapped by starting a separate project. However, the individual projects cannot currently be threaded together. For example, PBB therefore appears separately in both projects for communication with the supplier. However, this step

could be performed simultaneously and only once in one PBB. The following graphic displays the request in visualized form.

The PBBs highlighted in blue are merged into one work step on the basis of the operation data model. Consequently, resource times could be saved.

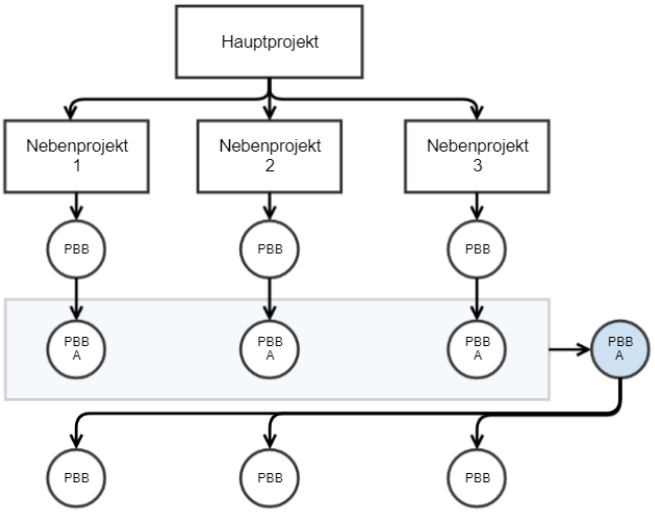


Figure 7 Cross Projecting

Access to other processes is also desirable. For example, in the case of a Diskover rollout during a fit-gap analysis, a functional extension of the software is often desired by the department. However, this extension must be anchored in the so-called core system. Consequently, the entire system needs a group-wide update after the implementation of this feature. The bottom line is that this task involves two separate projects. The first is the rollout project itself, in which the person responsible must be able to see whether his gap has already been closed. The second is the Diskover Development Project, in which the responsible person from the IT department needs an overview of the various requirements for the tool. PBB moves from A to B and back again. (Mirroring of the PBB in different projects)

The tool should be able to link projects to each other so that the processes can communicate with each other and avoid redundancies:

Example 1:

Business department specifies a requirement, but this must be mapped in a separate product management project of the IT department:

→ the IT department needs the product management view of the system

→ the department needs the development status of the individual requested components of the system

Example 2:

A large order for the customer is implemented - consisting of several projects. However, certain steps, e.g. sending the information to the supplier, can be done once, i.e. they can be combined.

The necessity of a "Project Consolidation" is especially striking in the service context. For example, a customer sends in a defective lawnmower, initiates a parallel enquiry in the call center and then orders accessories for his angle grinder.

Now the system should be able to recognize that it is *a* specific customer, even though different processes from different domains have been initiated and to bundle the return shipment of the repaired lawnmower and accessories to the same time.

5.1.1.9 Schedule

5.1.1.9.1 Forward calculation

Currently, only forward calculation from the start date of the project is possible. Based on predefined process time, a project end date or intermediate dates of e.g. milestones are calculated. This option should be retained.

5.1.1.9.2 Backward calculation

It should also be possible to plan the time backwards. For advertising campaigns, for example, flyers or similar must be ready by a certain end date.

5.1.1.9.3 Minimum and maximum times of operations

Each PBB should have a maximum, minimum and average process time. Alternatively a Mean and a standard deviation.

5.1.1.9.4 Consideration of resource availability

The availability of resources should be taken into account in time planning. The PM tool must know whether the resource is, for example, a part-time employee or already booked up for PBBs and then forward it to free resources. It must also be possible to map vacation days and substitutions, for example in connection with the resource time management system of EAG "Bedatime".

Resource loads and availabilities should be considered in future versions. Availabilities could be loaded from HR Management Systems (Holiday Planning, local public holidays, …) load scheems could be calculated. The load could then be used to expand or shorten expected process times.

5.1.1.9.5 Public holiday calendar

The PM-Tool should be able to manage public holidays and take them into account in the process flow, for example, public holidays in the Asian region such as Chinese New Year. Alternatively they could be imported from HR Management Systems.

5.1.1.9.6 Critical path

The new PM tool should be able to specify a critical path in the project sceleton. In this way, the future project manager of a process should be able to monitor the project even better by having the particularly critical processes controlled by the tool on his "monitoring board" in advance.

5.1.1.9.7 Project postponements

A simple model for rescheduling projects in case of project postponements. If, for example, a project is preferred, the project manager responsible should also be informed that this will affect project X, Y, Z and that these projects will then be postponed in their end date. The goal is to keep the utilization of project resources at the same level, even if a project is postponed.

5.1.1.10 Extension of the Capability Management

In the past, a test stand in the QA department had to be shut down when an employee left the company, because at that time no replacement skills for operating this machine could be found. Conversely, this also meant that the affected process module became obsolete. For this reason, a process building block should in future have a main owner and a co-owner in addition to the assigned roles.

The idea behind a “main owner” would be to regularly check his availability in the organization as well as to institutionalize checking the operativ success of PBBs enactment, training requirements for staff etc. at the owner. If the main owner is not available anymore, the co-owner changes to the main owner. This has the advantage that the process building blocks always remain executable. Who has been trained for a PBB is already documented. However, it is not possible to tell today when the user was trained for the PBB and how often he has already processed it. By carrying this information in combination with the frequency of execution, an overview of the existing competence levels for the individual PBBs would be possible in a simplified way (see point 5.1.1.3).

5.1.2 System actions

5.1.2.1 General notification functions and task list

Closely related to requirement 5.2.1.2, the system should generally send a notification to the user when a task is due – in particular for users that do not daily operate the PM Tool. Currently, this is controlled by a task list, which the user must call up specifically to see the tasks. Furthermore, new

tasks should be highlighted visually and in real time. For example through a pop-up window (cf. Outlook, Skype or similar).

If tasks are also shifted due to postponement of the entire project, this should also be made clear to the user in his task list. The detailed communication could be transmitted by the project manager via a message box (e.g. reasons for postponement). Blogging Funktion/Message Board Function/Dashboard.

5.1.2.2 Automatic alerts for tasks that are "overdue

Currently, overdrawn milestones turn "red" when the specified time has been exceeded, orange when they are due in the next 14 days and black when everything is "on time".

However, this only provides visual support for the project manager. He still has to send out "reminder" mails to the person responsible for the task to ensure that the task is processed.

In the new tool, the user should be able to recognize from the start, by accessing his e.g. daily dashboard, if tasks are in danger of falling behind. (Keyword: Automatic Monitoring / Controlling)

5.1.2.3 Time recording along the project

The new system should be able to document times during the processing of PBBs and thus also determine relevant key figures.

Example: process times, efficiency, degree of utilization, performance, etc.

5.1.3 System Interactions

5.1.3.1 Simplified management view of projects Create

Similar to the Matrix-Views mentioned above, a customizable monitoring should be made possible. A "Project Dashboard" will provide better visualization and, for example, be able to display diagrams. The project dashboard should customizable by/to the user. The user can then display the required process key figures in various forms.

- Schedules for presentations
- Project overviews /- portfolios
- Reporting key figures
- Number of open tasks e.g. per person/ per department

5.1.3.2 Checkpoints and rating of PBBs

For learning the system, it is also important that the user can give ratings for the whole process or for individual PBBs. Based on this, the system should propose optimization potential later on.

5.2 Non-functional requirements

5.2.1 Quality

5.2.1.1 Performance - faster system response times

Current loading times from the system, especially the matrix view (= tabular project overview), are clearly too long. Depending on how many values are displayed, the load time is 30-90 seconds. A generated Excel output even takes up to 15 minutes.

Especially in the subsidiaries Einhell China and Ozito, loading times are above average and not acceptable.

5.2.1.2 Standardize accesses

The future system must have the same access requirements throughout the Group. The subsidiary in Australia currently accesses the system via a remote desktop. Massive performance losses are the result here.

5.2.1.3 Changeability & maintainability

The system is to consist of independent modules so that individual modules can be added or removed at any time, thus ensuring the possibility of further development.

5.2.1.4 Reliability

The availability of the system is to be standardized throughout the Group. Currently, there are transmission problems due to multiple databases and mutual "locking" by users, which can subsequently limit the usability of the system for up to several minutes. Such weaknesses or even failures are to be avoided completely for the new system.

5.2.1.5 Independence + Transferability

It should be possible to use the system in different environments and only adapt the application to them. (e.g. iOS, Android, Windows versions, etc.) The installation effort should be as low as possible.

5.2.2 Ergonomics

5.2.2.1 Language

The leading language of the system shall be English. Nevertheless, the concept should be designed so that new languages could be added.

5.2.2.2 Improved communication within projects

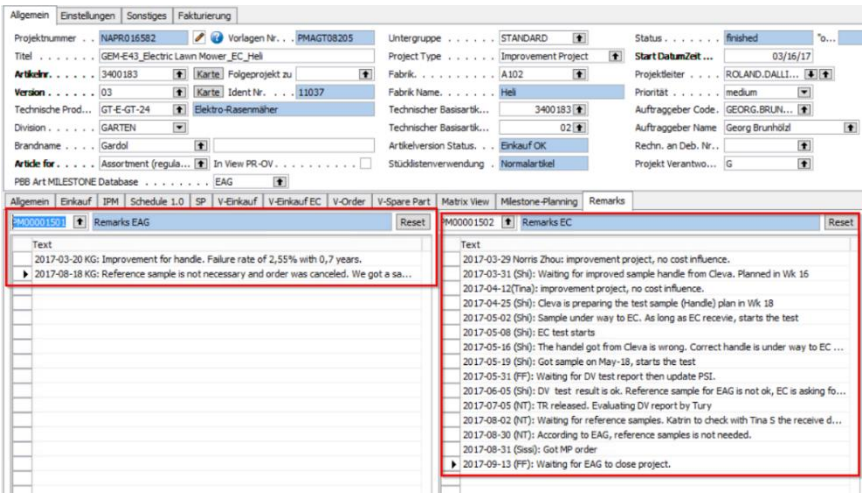


Figure 8 Current communication option via "Remarks

Currently, project documentation is made possible for users by so-called "remarks"

The requirement is to create an online chat facility that allows parallel "Live" communication with all participants, whereby either individual project participants can be contacted or several participants can be invited to a discussion topic about the project.

5.2.2.3 Global search function

A central search function is intended to support users in various searches for past and significant information content.

5.2.2.4 Web-based

The current PM-Tool runs on two databases. China and Germany are separated, but transmission errors occur from time to time. For example, tasks are displayed as "in progress" although they have already been processed. This has an impact on the project flow and disrupts the entire editing process.

5.2.3 Introduction, use and support

5.2.3.1 Transitional possibilities: Data management and migration

- How is old data transferred to the new system?
- What are the requirements for data preparation?
- Keywords: acquisition, adjustment,
- Conversion, new entry, data creation

A solution concept is to be developed for transferring already running projects or processes from the old system to the new one.

5.2.3.2 Securing long-term support / services

The later use of the new tool should be supported by a long-term support which the respective users can fall back on, e.g. the establishment of a helpdesk.

5.2.3.3 Online user documentation

Currently, the user documentation is stored in different places in different forms, such as Word, Power Point or Confluence

The documentation as well as user tips should therefore be made available online and in a standardized form, the same applies to short user videos.

5.2.3.4 Software documentation

For the continuous maintenance of the system, central documentation must be kept for both the design and the further developments.

5.2.3.5 Usability: understanding, learning and operating the system

Current main criticism (after various discussions):

- User guidance and operation of the system little or not at all intuitive
- Complicated menu navigation
- Not clearly arranged

Learning time for the current system:

- New employees: 1-3 months
- Existing employees (who are already working in the ERP system): 1-2 weeks

Based on these findings, it is absolutely necessary to fundamentally improve the usability of the system in order to understand the tool more quickly and learn how to operate the system more quickly.

5.2.4 Modifiability

5.2.4.1 Updates by users

Similar to app reviews, the user should be able to report bugs and send suggestions for improvement, which will be checked and processed in a regular cycle.

5.2.4.2 Communication of updates

If system updates are carried out, for example the completion of a new functionality, these should be actively communicated to the users.

Example: Update WhatsApp Messenger

Figure 9 Communication of Updates: Example WhatsApp

5.2.5 Legal, regulatory framework

Legal and regulatory frameworks must be specified during the detailed concept phase with the support of internal legal advice.

Above all, the collection of sensitive data, e.g. from the personnel area, must comply with the given legal situation.

Similarly, important document retention periods of, for example, supplier contracts, patents or the like must also be taken into account.

PART IV: Knowledge Management: Managing Organizational Intelligence And Knowledge In Autopoietic Process Management Systems – Ten Years Into Industrial Application

A. CIRP-CMS 2017 Paper

Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 00 (2017) 000–000

www.elsevier.com/locate/procedia

The 50th CIRP Conference on Manufacturing Systems

Knowledge management: managing organizational intelligence and knowledge in autopoietic process management systems – ten years into industrial application

Markus J. Thannhuber[a], Andy Bruntsch*, Mitchell M. Tseng[b]

[a]*Einhell Germany AG, Landau a. d. Isar, Germany*
[b]*International School of Technology and Management, Feng Chia University, Taichung, Taiwan*

* Corresponding author. *E-mail address:* abruntsch@connect.ust.hk

Abstract

A new approach to knowledge management in engineering domains was presented in the CIRP General Assembly 2001 with the title "An Autopoictic Approach for Building Knowledge Management Systems in Manufacturing Enterprises" [1]. Based on this a new process management system was developed and deployed. Today the system supports day to day engineering work of more than 300 engineering related staff on three continents. It drives organizational behavior by mimicking intelligence and the acquisition of knowledge, using both to derive suitable processes. This paper reports on lessons learned and may shed some light on future developments of knowledge-based manufacturing systems.

Peer-review under responsibility of the scientific committee of The 50th CIRP Conference on Manufacturing Systems.

Keywords: Knowledge management; Process; Organizational intelligence

1. Background

Key to the work presented here was a new understanding to and a new concept for intelligence and knowledge on an organizational level proposed by Thannhuber, Tseng and Bullinger [1] in 2001 and Thannhuber [2] in 2005. Up until this point knowledge management in industrial applications focused on acquiring data and information as well as managing them together with contextualizing meta information in IT Systems and delivering the right information at the right time to individual employees for their decision making or to support their value adding tasks. While these activities undoubtedly deliver their benefits however a new complementary approach was proposed that should better utilize efforts spent in the real world and potentials made available by phenomena that are described by 'intelligence', 'knowledge' or 'cognition'. Instead of purely empowering the individual in an industrial organization the focus should be shifted to the organization itself and how its behavior, responsiveness and efficiency can be improved by organizational knowledge and frameworks that breed intelligent behavior.

1.1. Phenomenological discussion of knowledge and intelligence on an organizational level

Although knowledge and intelligence are widely discussed for the human domain they are phenomena that evolution brought to emergence for natural systems in general to succeed in the competition to best adapt to their ecological niche and to best exploit the resources within it. Knowledge and intelligence help natural systems to derive successful behavior. Looking closer this 'successful behavior' in particular needs to balance three mutual exclusive abilities at highest levels: precision in execution wasting a minimum amount of internal resources while at the same time being able to cope with increasing dynamics in environments that constantly change and being able to cope with an increasing complexity. A situation well known to the engineering domain, too! Constituted out of

Peer-review under responsibility of the scientific committee of The 50th CIRP Conference on Manufacturing Systems.

human beings, cooperating to exploit marked niches, industrial organizations are nothing else but natural systems themselves [2].

Knowledge and Intelligence belong to the fundamental mechanisms to derive successful behavior. In the industrial context an organization's behavior is derived from coordinated activities its human staff develops by 'enacting' organizational processes. It is the sequence of activities, their coordination and interplay that leads to a company's successful behavior.

Intelligence in this context is a framework that enables an organization to derive well-coordinated and effective processes to behave responsive and successful. A framework that is built upon a suitable structure (physical infrastructure, components, … as given by IT systems, the hierarchy of command, process planning departments, etc.) and its organization (defining the interplay of the components as given e.g. by procedures on how to coordinate activities to a process). This processing framework allows an organization to take in stimuli, derive a suitable response process, support its enactment and capture the proceedings and success of the enacted process.

Knowledge is the content gathered or established in the processing framework based on which the framework's organization derives the assembly of activities to a suitable process. It is the content that drives intelligent processing and defines how the processing framework is further developed.

Knowledge management on an organizational level now gets a rather differentiated notion. There is no knowledge without intelligence! Managing knowledge on an organizational level first of all requires the management of intelligence. Managing knowledge in addition is all about deriving the right coordination of activities of staff members rather than increasing the individual knowledge of a single staff member.

1.2. Prerequisites for managing intelligence and knowledge on an organizational level

Earlier thoughts throughout the research on this topic suggested that there are a few necessary prerequisites to a working industrial implementation [1].

First of all system intelligence is to be institutionalized by a suitable processing framework. In today's industrial engineering context, systems institutionalizing the setup and enactment of industrial or engineering processes are 'Process Management Systems' (PMS). These systems typically come along side with modern Enterprise Resource Planning (ERP) or Production Planning Systems (PPS) or are complementary standalone solutions next to ERP and PPS. It is indicated that a suitable IT-based 'PMS' is an essential part of the processing framework.

In order for a 'PMS' to play an integral role in the described framework that implements the organization's intelligence it needs to provide a conceptual solution for Taylorization. Meaning that it needs to allow processes to be regarded as compositions of work units or work steps, which are the building blocks of any process. The system needs to be able to model them as generic 'Process Building Blocks' (PBBs) [1][2] that implement capabilities of staff members and that can be assembled to form different industrial processes. They are part of the system's structure. In order to support a reasonable assembly of building blocks (leading to a reasonable coordination of work steps) the PBBs need to be equipped with contextual information (descriptive data, wrapping information) which forms a part of the system's organization. The processing framework then needs to implement what was called 'declarative processing' [1] in other words it needs to be able to link up building blocks on demand and create a workable process instance by using the contextual information of PPBs against a given situative context.

We learned that suitable processing frameworks of intelligent systems require that available constituents of the system, the system's structure (PBBs, etc.), are permanently rebuilt. This does happen by incorporation of new capabilities, the encapsulation of complex procedures consisting out of several PBBs as one new single PBB (internalization), the depreciation of existing PBBs, and other 'deriving transformations' [1][2].

So does the system's organization, defining the interplay of all structural elements, mainly given by contextual information such as sequence information, rules that branch or control the invocation of PBBs, descriptive information and the like. In intelligent systems the organization, too, is permanently altered, adapted and rebuilt. [1][2].

The PBBs as the fundamental structural elements as well as their contextual information define the contents of the processing framework. They encode the knowledge of the organization that holds the processing framework. Knowledge management on an organizational level is the effort to promote this permanent acquisition of structural elements, new PBBs und contextualizing Information, the permanent evaluation of their effectiveness in operation as well as their adaptation and refinement thereafter.

This imposes high demands on the framework that should support the organizations processing. It must implement the possibilities to select and thus assemble PBBs in a suitable way as response to a situative context. This, being a process by itself, should be implemented self-similar [1]. From a system theoretic perspective a framework that permanently reproduces its structure and its organization is not just any framework, rather is it a highly special system. A System for which the cognitive biologist Maturana in 1972 coined the term autopoietic system, as a system that has the ability to generate its specific constitution – its components (structure) and their interplay (organization) – on its own [3]. In contrast to usual system definitions in the engineering world, where the system is an arbitrary set of elements, an arbitrary domain or space separated from the rest of the world by its boundary which is setup freely by the observer who intends to describe certain principles or theories for the system, for autopoietic systems the boundary is not up to the observers definition. The system rather is defined by all those constituents that are required to implement its autopoietic operation. They are real systems, just like the processing framework that implements the organization's intelligence.

2. Managing organizational intelligence and knowledge in the real world

During the years 1999 and 2001, when the conceptual approach was worked out, we developed an IT-based solution for the processing framework, called GCEN, to demonstrate and visualize our thoughts. At that stage we spent most efforts to realize an autopoietic processing framework that should be self-sufficient, modelling industrial engineering problem domains by modelling capabilities in PBBs and modelling all necessary functions to do the process assembly with PBBs, too. Autopoiesis this way was limited to a virtual world. The environment implemented a declarative processing approach. It was built up on a state-of-the-art agent-based system, introducing 'Believe-Desire-Intention' (BDI) reasoning, with each agent modelling a PBB. Processes where assembled by instantiating PBB agents that lived throughout assembly and enactment of a process as an agent community. The situative context as well as work-step results were acquired and presented for reasoning by the means of generic key-value-pairs (KVPs) built on a simple ontology. Being suitable for the 'lab environment' and demonstration purposes, difficulties did arise however when it came to real world engineering domain situations. In Complex real world situations e.g. the KVP structures and contents would not be sufficient for a successful BDI reasoning and declarative process assembly.

2.1. Building a new autopoietic processing framework

For its first industrial application we needed a new physical implementation of the autopoietic processing environment. Knowing of the importance of a proper process management we first built a new process management system based on the company's existing ERP System. It should provide the core functionality to the autopoietic processing framework similar to e.g. genetic encoding and genetic production of biostructures in living systems – just extremely simplified. Yet, we knew that this would not be the autopoietic framework itself but rather only a subsystem or functional part of it, as without the principles of self-containment, self-reference and the power of Artificial Intelligence (AI) the proclaimed permanent refinement and rebuilding of structure and organization would not be possible in an ERP-based process management system. A new approach to implement the true system scope would therefore be necessary.

The implemented process management system, which was simply called 'PM Tool', would provide all fundamental functionality to model, assemble and execute engineering processes. It caters for the modelling of staffs capabilities in PBBs, implements a KVP based documentation environment reflecting the real world at a given situation and capturing work progress as well as achieved results along the execution of work packages (PBBs), it guides the enactment of an assembled process (the execution of PBBs in the right sequence), it cares for resource assignment to executing PBBs, maintains 'To-Do'-lists for all staff members, and so on. In addition it does provide useful standard functionalities known from other process management systems such as records of true enactment sequences, process times or resource times and it allows for process or resource standard times to be pre-set to calculate typical engineering performance parameters such as utilization, resource performance, and the like. These allow the success of an enacted process to be evaluated and thus provide valuable input for structural and organizational refinements in the autopoietic system.

Unlike the original approaches implemented at GCEN, where process alterations during execution time always triggered a new declarative assembly that built a complete new process, in the PM-Tool new process control mechanisms were implemented that can alter the execution flow by selecting alternative pre-assembled process branches based on an evaluation of the situative context and/or execution results [4].

In GCEN most of the organization was modelled as contextual information, rules to be used to reason on the situative context, descriptive targets etc. all of which needed to be expressed on a high level of abstraction in order to enable declarative processing. In the PM-Tool we later took a much simpler approach mainly focusing on sequencing information, follower and predecessor or information on needs of sequenced or parallel execution of PBBs. Most of this contextual information to PBBs is modelled in simple tree-structures. In addition we introduced interfaces to subsystems that would provide own data structures to support the system's organization in a broader sense.

The biggest tweak of the new approach taken however was to identify a different scope of the autopoietic system itself. With GCEN every effort was taken to limit the system's scope tightly. We tried to confine autopoiesis to a virtual world in which structure and organization was permanently rebuilt in a purely digital manner. Employees, management staff or in general any human interacting with the system was regarded as an external resource, providing to or consuming from the system. They would not directly influence the declarative assembly of processes.

This has been changed for the implementation of the PM-Tool. Now staff members can be directly involved in the assembly of processes. They are now embedded in the systems organization – the new system is built with them rather than around them. The autopoietic process management system as a whole is now 'PM-Tool' PLUS 'human staff' with the PM-Tool playing the role of a defined part only. Next to the digital organization now human organization blends in and involves in process control, preparation of contextual information for process assembly and providing PBBs. Being natural real systems themselves this does not contradict the theoretic concepts but it rather establishes a tighter link between system levels [2].

2.2. First application field: Lab operation

The first real world industrial application of this new approach to manage intelligence and knowledge on an organizational level was the laboratory operation of Einhell Shanghai. The lab does testing and qualification of consumer power tools and power gardening equipment - both electrical and petrol driven items. The challenge here is that the process to be enacted is unique for every sample to be tested in the lab. Every product has a different technical structure implying

different technical aspects to be tested to ensure a proper quality performance in field later on. For safety aspects the products need to be tested against numerous national and international standards. And even samples of the same product won't be tested along identical test processes as tests are allocated to different samples.

The processes to be enacted are all the test procedures sequenced to a work process for a single sample. The process building blocks (PBBs) are the single tests to be carried out. The challenge in this context is selecting the right tests, sequencing these tests correctly without influencing results of later tests, allocating the right tests to the right samples, collecting test results in a structured way, evaluating test results against defined verdict criteria and last but not least a suitable reporting. All of this is typically carried out manually in the hands of experienced management staff and assigned test engineers.

We have been introducing the process management module (PM-Tool) in 2006 transforming the lab operation in a test factory. The easy part was modelling the PBBs as they represented well known tests, the capabilities of the lab and its staff. The PBBs contents typically had to cater for Standard Operating Procedures (SOPs), descriptions of test parameters to be adjusted and test result parameters to be collected all of which was achieved by the generic KVP environment provided by the PM-Tool. The difficult part was the organization of the autopoietic system driving the assembly of workable test processes and ensuring their completeness. For this purpose the PBBs where wrapped in contextualizing information describing their typical application in relation to other PBBs or in the context of demands typically posed by the situative context. They were organized in little branches that typically ensured a suitable sequencing while the branches belonged to schemes or templates that are selected based on the projects context such as the product type of a sample defining its needs to proof compliance. In addition a Failure Mode and Effect Analysis (FMEA) system was established that evaluates in field data against given product structures and proposes dedicated test. It is driven by statistical models delivering unique suggestions at any point of time and for any given product structure. Both approaches are used for the assembly of tailored test processes for every lab sample.

Since the introduction in 2006 more than 16500 projects with 25500 samples have been carried out. We have been executing roughly 300000 test PBBs to finish qualifications acquiring almost 2 million physical results. Today the lab operates in a process landscape maintaining KPI-sets such as process performance, utilization, primary and secondary processing times, etc. well known in industrial engineering but untypical for lab operations. Aside from optimizing the industrial performance the consistency and overall quality as well as the diversity of the qualification work could greatly be improved.

2.3. Second application field: Design, engineering and development

The second industrial application of the 'PM-Tool' was the domain of design, engineering and development where technical projects would be developed to marketable products. By nature design and development are characterized by the uniqueness of their contents, the unpredictability of whether technical solutions prove themselves feasible and with-it the uncertainty about engineering challenges that need to be sorted out on the way to a successful product. Again this is typically a project-organization's world. Yet it was clear that certain proceedings are more successful than others and that certain work packages and sequences are mandatory for successful projects.

In 2007 we have been introducing a systematic process management with the PM-Tool and transferred the design, engineering and development operation into a project factory. In this case modelling the initial sets of PBBs was a bigger challenge than introducing the right organization. Representing the work packages along technical projects it was difficult to find the right level of abstraction and granularity. Lifting the PBB modelling onto a higher level of abstraction is necessary as the particular content of a work package, its microscopic execution, is very much driven by the technical context, its problem and solution domain – which would be hard or even impossible to model. On the other hand there is a target or goal to every work step in a development project that fits the work package into its larger context. To derive a proper PBB modelling we focused on these abstract targets and their descriptions, the parameters that would define under which conditions work packages are to be carried out and the parameters that would describe what was achieved (results). After some time of experimenting we found a proper approach to model with the right degree of abstraction and in the right granularity – from then on newly built PBBs replacing others further improved these aspects.

The organization of the autopoietic system was implemented based on a similar approach as already in operation in the test factory: The PBBs were wrapped in contextualizing information that would organize them in branches which ensured a suitable sequencing while the branches belonged to schemes or templates that are selected based on the projects' situative context. This time however the size of schemes or templates was of a different order. A typical technical project enacts a process with 900 to 1100 work packages. To gain efficiency in the process assembly the templating functions were improved allowing the preparation of alternative process branches upfront. In addition, process control mechanisms have been introduced in the PM-Tool to allow (semi-)automated selections of branch alternatives by evaluating execution results documented in the KVP-framework.

Since 2007 we have been enacting more than 10500 projects developing just as many products or product modifications. Along with the technical department others such as the purchasing, product management and marketing have joined the system. The system today operates extending across locations in Europe, Asia and Australia. Today more than 330 users directly engage with the system having executed more than 1.3 million PBBs.

R&D, engineering and design are a typical project oriented domain, with engineers driven by virtues like creativity, flexibility and solution orientation the introduction of a process

world is typically a very difficult if not impossible task. Engineers in this domain are used to project management and they feel they lose responsibility and trust if they have to operate under process regimes that they do not directly control. Introducing the PM-Tool we faced exactly this challenge. After a while in operation however the engineers found they are relieved from coordinative workload leaving them with more time to spend on real engineering issues, more room for creativity and a better value added gained out of their daily work.

3. Results and learning from ten years of industry application

The biggest learning for us was that knowledge management on an organizational level requires to spend most efforts on modelling and managing the organization's capabilities and their interdependencies. Understanding them as the fundamental building blocks for the assembly of industrial processes it is crucial that organizing information is available supporting the right sequencing and selections.

Having moved the organization's human staff to the center of the autopoietic knowledge management system was essential as pure AI approaches for process assembly in an industrial world would have been a too big and too risky step to take. The defined integration of human interaction in the process assembly and definition as well as in monitoring and execution control allowed staff members to sit in the driver seat while having superb drive assistance systems in place to guide their activities and limit their efforts to a minimum. This lead to a good system acceptance.

Coordinated processes in the context of the presented approach are a natural result of the mechanisms implemented with the autopoietic framework. Instead of leaving procedural enactments completely open and just documenting their results as typically done by ERP systems and instead of tightly restraining processes as in workflow systems, processes now are just a result reflecting the situative context within which they are enacted. They are 'per se' agile and do not need to be artificially 'agiled' as in workflow systems. This way the processes naturally work in industrial contexts, not only over a few process steps but are enacted in 900 to 1100 steps as typically experienced in R&D and design engineering projects. Projects enacted in coordinated processes and executed under the regime of the PM-Tool process management environment relieved the engineers from coordinating activities which we estimate to have occupied as much as 30% of their worktime.

But was knowledge and intelligence really applied to the benefit of productivity and leverage the company's profitability? A question that needs some thoughts to be answered as annual comparisons are difficult with engineering projects strongly depending on the types of projects, products and context under which they are carried out. But intelligent organizations, systematically acquiring and applying knowledge, should be able to cope better with complex environments or in other words complex contexts. A paradigm suitable for evaluating the interaction of processes and their respective context has been developed by A. Bruntsch [5]. It contrasts the dynamics inherent to processes and their corresponding context in terms of measured entropies. While in its physical origin, entropy quantifies disorder in a system or the uncertainty of a system's state [6, ch. 5], process entropy H_p has been derived as a measure of variation in process enactment [5, ch. 4]. A static process that is repeated identically over time thus apparently implements zero entropy. Yet, the more frequent a process is reconfigured dynamically, the larger H_p gets. Process context entropy H_c, analogously, has been introduced for quantifying the diversity in contextual situations that a process operates in.

By applying this paradigm using actual process data of activity-level granularity we were able to measure how both entropies developed over time (see Fig. 1). We found that the entropy H_c of the processes' context, as a measure for the complexity of the organization's environment, increased on average at a rate of 1.5% p.a. over the years 2009 to 2012. This can be explained by rising customer expectations as well as increasing demands to demonstrate compliance against national and international regulations and standards. Notably, at the same time the entropy H_p of processes, representing the complexity of project executions, decreased with remarkable 7.5% p.a. under the PM-Tool regime [5, ch. 12]. In typical R&D environments one would expect that the more complex projects get, the more complex their realization becomes, thus implying a positive correlation between both entropies. Here, in contrast, the findings indicate that increasing contextual complexity was handled by an even decreasing internal complexity. Hence, the organization was able to cope with a more difficult environment by applying organizational intelligence and acquiring organizational knowledge that allowed simpler internal processes.

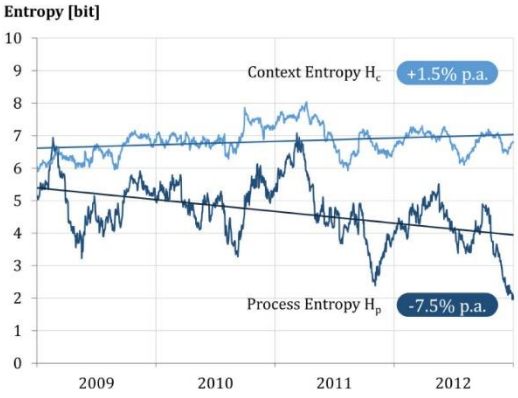


Fig. 1. Despite rising contextual complexity, process complexity could be significantly reduced over time (adapted from [5, ch. 12]).

By limiting the scope of analysis to projects with a rather uniform characteristic, decoupled from external complexities, the resulting actual performance impact can still be uncovered. An exemplary investigation of 163 solely new product development projects, finished between 2010 and 2015, revealed a significant annual reduction in average process and resource times of 13.1% and 16.4% respectively (see Fig. 2). This clearly indicates the effectiveness of the autopoietic

approach to process management.

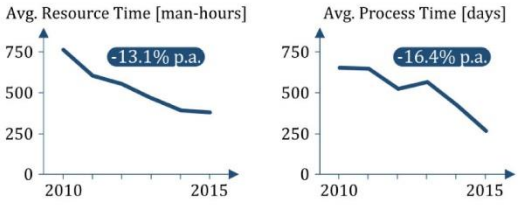


Fig. 2. Effectiveness of the autopoietic process management approach revealed by significant performance improvements.

Aside from numeric performance measures the biggest benefit of the system, however, was of an intangible nature. In times were staff fluctuations become a critical factor the ability of an organization to integrate new employees and get them productive is highly important. We found that the introduction of the PM-Tool with its integrated process management system greatly improved our ability to kick-start newcomers. They feel more comfortable in a network of well-coordinated processes were they can focus on their expertise in well-scoped work packages instead of having the need to coordinate with people or an organisation they hardly know.

4. Conclusion

It is never easy to introduce new scientific concepts to the industrial world in practical operation. Many hours were spent for implementation and even more for persuasion. Today, however, we could not imagine to operate without the processing environment “PM-Tool”. It proofed to help us with the growing complexity. It helps us to stay dynamic and efficient. Knowledge Management is now happening within and is applied to our operation and is not something to be done “Add-On”.

References

[1] Thannhuber MJ, Tseng MM, Bullinger HJ (2001) An Autopoietic Approach for Building Knowledge Management Systems in Manufacturing Enterprises. CIRP Annals – Manufacturing Technology 50(1):313-318.
[2] Thannhuber MJ (2005) The Intelligent Enterprise, Physica/Springer.
[3] Varela F (1979) Principles of Biological Autonomy. New York: Elsevier (North Holland), p. 13
[4] Bruntsch A, Tseng MM (2016) An Approach for Process Control of Responsive Service Processes. CIRP Annals – Manufacturing Technology 65(1):459-462.
[5] Bruntsch A (2015) The Fundamentals Behind Responsive Processes (Doctoral dissertation). Retrieved from http://lbezone.ust.hk/bib/b1514431
[6] Sethna JP (2006) Entropy, Order Parameters, and Complexity. Oxford, New York: Oxford University Press

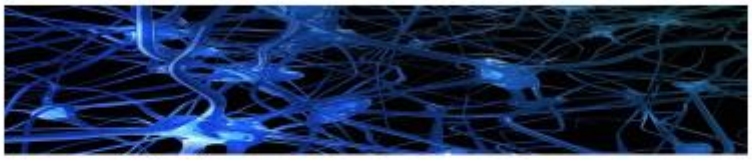
B. CIRP Manufacturing Systems Knowledge Management - Presentation

„Knowledge Management" in a classic sense:

- Acquiring Data and Information (Contents)
- Contextualizing Contents with meta information
- Delivering the right contents at the right time to employees

→ **empowering the individuals**

Knowledge and Intelligence from a wider perspective

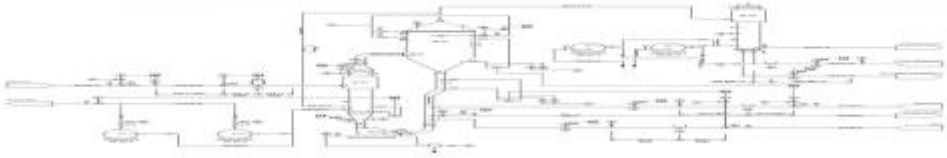


„Knowledge" and „Intelligence" as evolutional Phenomena:

- Help natural systems to derive successful behavior
- To best adapt and exploit their ecological niche
- Enabling **precision** of execution, coping with **dynamics** and **complexity**

→ **empowering the System (Organization)**

Knowledge and Intelligence in the Industrial Context



Successful Behavior of an Organization:

- Is derived by coordinated activities of human staff and operated equipment
- In organizational processes that deliver results

Intelligence is the framework that enables on organization to derive well-coordinated and effective processes to behave responsive and successful.

Knowledge is the content gathered within this framework, enabling the successful assembly of activities to a successful process

Knowledge and Intelligence in the Industrial Context

Successful Behavior of an Organization:

- Is derived by coordinated activities of human staff and operated equipment
- In organizational processes that deliver results

Intelligence is the *framework* that enables on organization to derive well-coordinated and effective processes to behave responsive and successful.

Knowledge is the content gathered within this *framework*, enabling the successful assembly of activities to a successful process

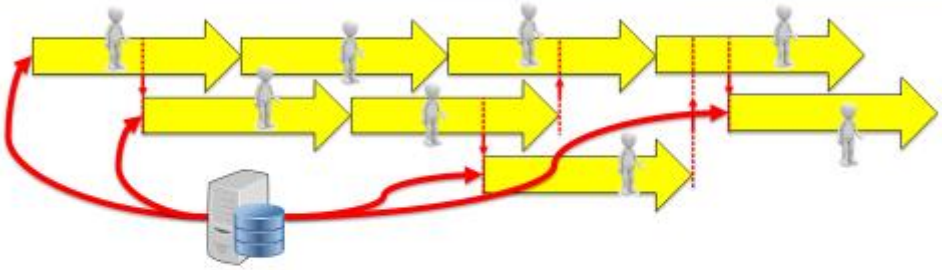
Intelligent Processing – Process Management to build the Framework

The Framework that enables intelligent processing

- Requires a suitable STRUCTURE (physical infrastructure, Components, … as given by an IT System, process planning departments, hierarchy of command, …)
- And an ORGANIZATION (defining the interplay of components, … as given by e.g. procedures on how to coordinate activities to a process)
- It allows to take in stimuli and derive a suitable response process

To institutionalize the management of Knowledge and Intelligence we proposed to introduce a **Process Management System** that implements this Framework.

Preliminary Conclusion for Knowledge Management in Industrial Organizations



Knowledge Management

- Is less about the individual but rather about the coordination of all human employees and equipment in processes
- Can't be thought without thinking about **managing intelligence**
- Needs a proper IT-based institutionalization in a **Process Management System**

Einhell GUT GEMACHT

7

First Conceptual Drafts 1999-2001: Process Management in GCEN

A First implementation of a conceptual Process Management System that should have been able to cater for the demands: GCEN

- GCEN allowed the **modelling of work units or work steps** that could be assembled as building blocks to a process -> conceptual solution for "**Taylorization**"
- These units, we called them „Process Building Blocks" (PBBs), each represent staff capabilities that can be assembled to form different industrial processes
- The PBBs form the **STRUCTURE** of the system
- To support the assembly to a process they need to be equipped with contextual information (descriptive data, wrapping information)
- The assembly itself was done by "declarative processing" where PBBs have been selected and joined up resolving targets and preconditions. A Process by itself.
- The contextual information as well as the declarative processes formed the **ORGANIZATION** of the system
- The System was implemented based on a BDI-Agent framework

The Processing Framework – an Autopoietic System to facilitate Knowledge Management

Suitable Approaches for the Process Management System

- allowed the permanent rebuilding of „Process Building Blocks" (PBBs) by incooperation of new blocks or encapsulation of successful but complex known procedures to single PBBs.

→ The STRUCTURE of the system is permanently refined / rebuilt

- In addition, based on the success of assembly and instantiations of processes, the contextual information (descriptive data, wrapping information) driving the assembly of PBBs to processes is permanently altered, adapted and rebuilt too.

→ The ORGANIZATION of the system is permanently refined / rebuilt

Maturity

Structure and Organization of a system being permanently rebuilt = **Autopoiesis**

With **PBBs** and their **Contextual Information** being the contents of the Processing Framework they are the **Knowledge** that is **autopoietically managed**.

Einhell GUT GEMACHT.

9

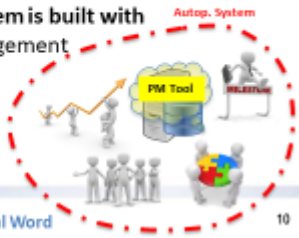
Process Management in the Real World: a Change of Scope

Where in the "LAB Environment" every effort was spent to build GCEN itself as a closed autopoietic system(self-contained and self-referenced), in the real word things were altered:

Lab Environment

An ERP-based Process Management System
The implemented process management system, called '**PM Tool**', would provide all fundamental functionality to model, assemble and execute engineering processes – however it was not self-contained autopoietic and thus only a part or subsystem of the processing framework.

Now staff members can be directly involved in the assembly of processes. They are now embedded in the systems organization – **the new system is built with them rather than around them**. The autopoietic process management system as a whole is now 'PM-Tool' PLUS 'human staff'.
A new scope!



Real Word

Einhell GUT GEMACHT.

10

PM-Tool (Real World) vs GCEN (Lab Environment): Technical Aspects

Technical differences between PM-Tool (Real World) and GCEN (Lab Environment):

PM Tool (Real World)	GCEN (Lab Environment)
• new process control mechanisms can **alter execution** flow by selecting alternative pre-assembled process branches based on situative context / execution results	• process alterations during execution time always triggered a new declarative assembly that built a **complete new process**
• Organization: focusing on sequencing information, follower and predecessor or needs of sequenced or parallel execution. Mostly modeled in **simple tree structures**	• Organization: contextual information, rules, descriptive targets etc. all expressed on **high level of abstraction**
• records of true enactment sequences, process times or resource times → **utilization, resource performance**, ...	• No Resource binding; no resource performance evaluation, utilization, ...

Einhell GUT GEMACHT.

11

First Field of Application: Einhell Qualification Laboratory Operation

What it does:

- The lab does testing and qualification of consumer power tools and power gardening equipment

Challenge:

- the **process** to be enacted is **unique for every sample** to be tested in the lab (different technical structures; numerous national and international standards; ...)

Einhell GUT GEMACHT.

12

First Field of Application: Einhell Qualification Laboratory Operation

The Approach:

- The processes are all the test procedures sequenced to a work process for a single sample
- The process building blocks (PBBs) are the single tests to be carried out

Target:

- selecting the right tests,
- sequencing these tests correctly without influencing results of later tests,
- allocating the right tests to the right samples,
- collecting test results in a structured way,
- evaluating test results against defined verdict criteria
- a suitable reporting.

Einhell GUT GEMACHT

13

First Field of Application: Einhell Qualification Laboratory Operation

Modelling the work packages (the easy part):

- modelling the PBBs as the capabilities of the lab and its staff. The PBBs were modelled to be the **single tests** to be carried out
- PBBs had to cater for Standard Operating Procedures (**SOPs**), descriptions of **test parameters** to be adjusted and test **result parameters** to be collected.

Modelling the organisation (the difficult part):

- PBBs where wrapped in contextualizing information describing their typical application in relation to other PBBs
- organized in little branches that typically ensured a suitable sequencing
- branches belong to schemes or templates that are selected based on the projects context
- A Interface connection to a FMEA-System was established that evaluates field data against given product structures and proposes dedicated test.

Einhell GUT GEMACHT

14

First Field of Application: Einhell Qualification Laboratory Operation

Results:
Introduction in 2006, since then:

- more than 16500 projects carried out
- with more than 25500 samples
- executing roughly 300000 test PBBs to finish qualifications
- acquiring almost 2 million physical result parameters

The successful setup of a "Test Factory"
Today the lab operates in a process landscape maintaining KPI-sets such as process performance, utilization, primary and secondary processing times, etc. well known in industrial engineering but untypical for lab operations.

Engineering Knowledge is maintained, derived and managed in the PM-Tool that controls the lab operation and leads to successful behavior.

Einhell GUT GEMACHT

15

Second Field of Application: Design, Engineering and Development

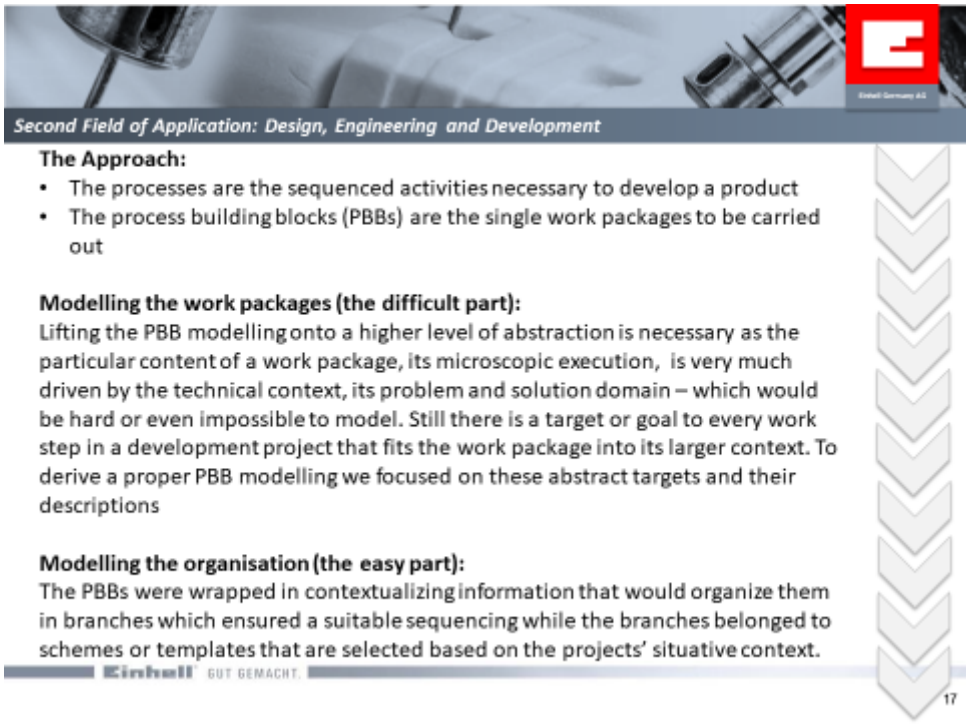
What it is about:

In the domain of design, engineering and development technical **projects are developed to marketable products.**

Challenge:
By nature design and development are characterized by the uniqueness of their contents, the unpredictability of whether technical solutions prove themselves feasible and with-it the uncertainty about engineering challenges that need to be sorted out on the way to a successful product.
A typical project-organization's world

Einhell GUT GEMACHT

16


Second Field of Application: Design, Engineering and Development
The Approach:
• The processes are the sequenced activities necessary to develop a product
• The process building blocks (PBBs) are the single work packages to be carried out
Modelling the work packages (the difficult part):
Lifting the PBB modelling onto a higher level of abstraction is necessary as the particular content of a work package, its microscopic execution, is very much driven by the technical context, its problem and solution domain – which would be hard or even impossible to model. Still there is a target or goal to every work step in a development project that fits the work package into its larger context. To derive a proper PBB modelling we focused on these abstract targets and their descriptions
Modelling the organisation (the easy part):
The PBBs were wrapped in contextualizing information that would organize them in branches which ensured a suitable sequencing while the branches belonged to schemes or templates that are selected based on the projects' situative context.
Einhell GUT GEMACHT.
17


Second Field of Application: Design, Engineering and Development
The Result:
Introduction in 2006, since then:
• more than 10500 projects carried out
• Along with the technical department others such as the purchasing, product management and marketing have joined the system
• The system today operates extending across locations in Europe, Asia and Australia
• more than 330 users work on the system
• more than 1.3 million PBBs have been executed
The successful setup of a "Project Factory"
The biggest challenge was to overcome the engineers emotions feeling the lose responsibility and trust if they have to operate under process regimes that they do not directly control. After a while in operation however the engineers found they are relieved from coordinative workload leaving them with more time to spend on real engineering issues.
Drawing
3D model
Sample (foam)
Finished 3D model
Prototype
Mass production
Einhell GUT GEMACHT.
18