Function-behaviour-structure model for modular assembly equipment

This item was submitted to Loughborough University’s Institutional Repository by the/an author.


Additional Information:

- This is a conference paper.

Metadata Record: https://dspace.lboro.ac.uk/2134/25336

Version: Published

Rights: This work is made available according to the conditions of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) licence. Full details of this licence are available at: https://creativecommons.org/licenses/by-nc-nd/4.0/

Please cite the published version.
FUNCTION-BEHAVIOUR-STRUCTURE MODEL FOR MODULAR ASSEMBLY EQUIPMENT

Niels Lohse, Svetan Ratchev, Anthony Chrisp

Precision Assembly Laboratory, School of MMEM, The University of Nottingham, Nottingham, NG7 2RD
Tel. +44 115 951 3875, Fax. +44 115 951 3800

ABSTRACT
Reconfigurable modular manufacturing systems provide a solution to manage current challenges of dynamic, customer driven markets. Powerful methods are needed for rapid configuration of system. This research focuses on the ontological definition of modular assembly device domain knowledge which builds the foundation for such methods. In this word formal representations will be defined based on linked models of functions, behaviour and structure of the equipment modules. The method will be discussed using an illustrative example.

INTRODUCTION
Driven by the need for mass customisation and increasingly shorter product life cycles, there is currently a strong trend towards the development of precision assembly systems that can be rapidly configured and reconfigured from standardised modules (Onori et al. 2002). This opens the scope and need for a computerised method to aid and automate the configuration process of such systems. Traditionally, the design process of assembly systems goes through three stages. It starts with the definition of the product which needs to be assembled, continues with the specification of the processes needed to assemble that product and finishes with the design of an assembly system which can deliver the required processes (Rampersad 1994). Substantial research effort has been dedicated to the planning of assembly processes based on product models. The design of assembly systems based on process requirements, however, has not been sufficiently explored. Only approaches on a very abstract level or with very limited scope have been proposed and there is a strong need for the development of more integrated and detailed approaches.

The first challenge, when approaching the definition of a formal method for the assembly process based configuration of modular assembly systems, is to define a formal data and information model that can be used in the design process and which objectively describes module capabilities to the required level of detail. The second challenge is posed by the formalisation of the configuration process itself, for example using algorithms. In the following we will concentrate on the formalisation of the device capabilities. However, this report is not focused on the configuration process itself, but it has been taken into consideration since it strongly influences how the capabilities of the modules should be formalised.

Two approaches can be identified to capture the capabilities of assembly resources: a description based on the processes they can perform and a description based on their behaviour and structure and their link to functions. Both can be mapped against the required process capabilities. However, the specification of capabilities purely based on precision assembly processes tends to be rather subjective. The definition of the functions, behaviour
and structure of the modules combine the user specific view of their capabilities with a more objective definition in terms of behaviour based on physical principles (Umeda et al. 1996).

Several approaches have been reported for the definition of device or module capabilities. Zha et al. (2001) use knowledge intensive Petri net for the modelling and analysis of assembly equipment and systems. A language representation of function-behaviour-structure for mechanical devices has been introduced by Sasajima et al. (1995) based on ontological engineering principles (Mizoguchi et al. 2000). Their main focus is on understanding the functional capability of devices based on their behaviour and structure. Umeda et al. (1996) and Tomiyama et al. (1993) use qualitative physics to define the relation between structure, behaviour and functions. Sasajima et al. (1995), Mizoguchi et al. (2000), Umeda et al. (1996), Tomiyama et al. (1993) define behaviour based on physical phenomena.

The focus of the research reported in this paper is on the description of assembly equipment modules using their functions, behaviour and structure. The first question that is being addressed is the clear distinction and definition of these key characteristics of a module. Secondly we introduce a formal ontology for the representation of functions and a joint representation of behaviour and structure. Thirdly we look at the application of this ontology to model equipment modules and finally we conclude with the question of how this can be used to configure assembly workstations and system based on required processes.

Figure 1 Conceptual diagram of the entities and relations in the data model

OVERVIEW
Before we address the actual requirements for and implementation of the function-behaviour-structure model, it is necessary to clarify the distinction between these three aspects of the model. The functions express the capabilities of a module based upon the intention of the designer and are therefore subjective. Functions are generally defined as an abstraction of behaviour for a specific use or purpose (Umeda et al. 1996). The assumption at this point is that the purpose of the module will be to assemble a product and all function will be interpreted accordingly. The behaviour of a module defines how it reacts to changes in its environment and in turn how this reaction changes the environment. Behaviour is defined as
state transitions from input to a module to output from it based on physical phenomena that provide the building blocks for an objective description of assembly equipment modules. The structure defines the physical model of the modules with objects, attributes and relations.

The first challenge when defining a function-behaviour-structure model is that all three aspects are interdependent. The data model has been defined in a way that there are entities representing the three different aspects of the modules which are linked to define their relationships (see Figure 1). Since the behaviour and the structure can be defined objectively, the link between these two is much stronger than between them and the functions of the module. The desired result is that the same modules can have different functions depending on their intended use, e.g. pliers could either have the function “hold something” or “form something”.

The second challenge is how to use definitions of the three aspects to reason about the capabilities of modules. A method for the mapping of different structural, behavioural and functional entities against the required process capabilities will be briefly discussed in the following sections. Since functions express the design intent, the higher level functions of a module intended to be used for assembly are equivalent to a part of the required assembly process. The assembly functions of a module can therefore be matched directly against the process requirements.

The design process imposes certain requirements on the model. First of all, the entities in the model with their attributes need to be individually traceable to be able to follow how certain aspects influence design decisions. Secondly, the three aspects need to be decomposable until they are built of the smallest aspect entities, so called base entities or atoms (physical objects, physical phenomena and base functions). This provides the means for design abstraction, which is commonly used to reduce the complexity a designer has to deal with at any one step during the design. The underlying data model has been developed based on ontological engineering principles. The fundamental building blocks are classes, instances and slots.

**FUNCTION DEFINITION**

Since this research is looking at the data formalisation for the assembly system design process, functions are defined as part of the process definition. The process here is seen as a number of assembly oriented functions executed in sequential order in time. The assembly process (P) of a product is therefore defined as a set of assembly specific functions (F) and a set of their precedence relations <PR>:

\[
P = \{F_1, ..., F_n, <PR>_{1, ..., <PR>_{n_r}}\}
\]

(1)

The precedence relations are defined as a sequential constraint between two functions, e.g. hold part A before moving it <PR>_{1} = (<before>, F_{hold}, F_{move}). However, the focus of this work is on the functions themselves and how they can be related to the behaviour of different components rather than on a complete process definition.

The functions are defined in the form of “to do something” based on a verb specifying the activity and objects defining the entities that are the focus of that activity. During the design process of a system designers decompose the required functions into sub-functions until they arrive at suitable base functions (Pahl & Beitz 1996) and (Roth 1982). Functions are therefore hierarchically structured in a manner whereby they become more specific further down the hierarchy. Each function can have a number of attributes, besides the objects, which define
the specific function. As discussed in (Ratchev et al. 2003), the process has three distinct levels that are imposed by the traditional structures of assembly systems: task, operation and action level. These correspond to the entities: workstations, devices and elements in the assembly system structure respectively. Each function is defined by its type \(<TY>\) and level \(<L>\), a set of objects \((O)\), a set of attributes \((A)\) and a set of sub-functions \((F)\) (see Figure 2). Additionally any function can be linked to a set of one to many modules \((M)\) to specify preferred equipment during the requirements definition.

\[
F_f = \{TY,<L>,O_1,...,O_{n_o},A_1,...,A_{n_a},F_1,...,F_{n_f},M_1,...,M_{n_m}\} \tag{2}
\]

All the sub-functions are also part of function set that defines the process. Attributes are slots and are therefore defined as independent entities linked to the function, which allows them to be individually traced during the design process. A function can have attributes defining positions, magnitudes, directions, orientations, etc. all of which are defined in relative terms based on the notions of qualitative physics (Forbus 1984). Objects come from three domains: energy, entity (all physical things except energy) and information (Kitamura et al. 2002) and (Pahl & Beitz 1996) (see Figure 3).

**STRUCTURE AND BEHAVIOUR DEFINITION**

Before addressing the structure of modules specifically, it is necessary to look at the entities that define a system in general. As previously discussed assembly functions deal with objects in the form of parts that are assembled into a product. Like the actual equipment modules of the system these are made from solid objects. On the behaviour modelling side there are other types of entities that need to be defined in order to describe the transition from energy and information to the assembly of parts. All objects are organised into categories and linked with sub-type/super-type relations that also work as inheritance for the attributes. Attributes are linked to the objects at different levels of abstraction (see Figure 3).

\[
O_i = \{\text{subtypeOf}, \text{type}, \text{description}, A_1,...,A_{n_a}\} \tag{3}
\]
The structure of a system is defined as a set of modules (M) and their connections (C) (Ratchev et al. 2003). Figure 4 shows the structure and behaviour of a 2 finger pneumatic gripper. Modules are physical objects and each module can have either sub-modules or solid objects that define its structure.

\[
O_1 = ("Pneumatic pressure-open", \text{gasEObj}, m_1); \quad I_3 = (2, \text{<female>}, PO_2); \quad I_4 = (3, \text{<female>}, PO_3);
\]
\[
O_2 = ("Pneumatic pressure-close", \text{gasEObj}, m_2); \quad I_6 = (3, \text{<male>}, PO_5); \quad I_7 = (4, \text{<female>});
\]
\[
O_3 = ("Finger translation", \text{transEObj}, m_3); \quad \text{PO}_2 = (\text{<input>}, O_2); \quad \text{PO}_3 = (\text{<output>}, O_3);
\]
\[
O_4 = ("Finger translation", \text{transEObj}, m_4); \quad \text{PO}_5 = (\text{<input>}, O_3);
\]
\[
O_5 = ("Gripper body", \text{solidObj, mat}, \text{geo_1}); \quad B_1 = (\text{PO}_1, \text{PO}_3, \text{<changeEType>>});
\]
\[
O_6 = ("Finger", \text{solidObj, mat}, \text{geo_2}); \quad C_1 = (I_4, I_6);
\]
\[
M_1 = ("Gripper body", O_5, I_1, I_2, I_3, I_4, I_5); \quad M_2 = ("Finger 1", O_6, I_7, I_8);
\]
\[
M_3 = ("Finger 2", O_6, I_9, I_10);
\]

Figure 4 Structure and behaviour definition of a 2 finger pneumatic gripper

\[
\text{Structure} = \{M_1, ..., M_{n_M}, C_1, ..., C_{n_C}\}
\]

\[
M_i = \{\text{description}, M_{i,1}, ..., M_{i,n_M}, O_{S,1}, ..., O_{S,n_O}, I_{1}, ..., I_{n_I}\}
\]

To constrain the explosion of possible structure configurations each module has a number of specific interfaces (I) that can only be connected to fitting interfaces of other modules \(C_i = (I_i, I_j)\). The definition of the interfaces is based on previous work by Ratchev et al. (2003), but has been extended to include ports \(<PO>\) that specify how objects can be passed between different modules (Sasajima et al. 1995). For the purpose of this paper a simplified definition is used to demonstrate the more general aspects of the model. The only interface attributes considered are a unique type \(<TY>\) and direction of the interface. These two attributes define whether two interfaces fit together. Only interfaces with the same type and corresponding directions can be connected.

\[
I_i = \{<TY>_i, \text{direction}_i, <PO>_1, ..., <PO>_n_{PO}\}
\]

The ports define which objects can pass through an interface into a module and which are coming out again. A port can therefore be described as an object and direction (input, output or both):

\[
<PO>_i = \{\text{direction}_p, O_i\}
\]

The equipment modules and equipment in general have two types of behaviour: passive and active. The passive behaviour of a module defines how the module reacts to changes that are inflicted upon it. For example, the mass of a module influences how the module reacts to forces. The active behaviour on the other hand, defines how a module can change other entities. For example, a motor changes electrical energy into rotational energy. The passive behaviour of a module is defined by variables that are associated with it and have been referred to as its attributes. The active behaviour will be defined as mappings of input ports of
to output ports based on physical principles. Active behaviour will be referred to as behaviour (B) only.

\[ B_i = \{ < PO >_{in,i}, < PO >_{out,j}, B_n, \ldots, B_m, < PP >_{pre} \} \] (8)

Since modules are composed from lower level modules or solid objects, the behaviour of these lower level structural entities needs to be linked to the behaviour of the higher level entity. This has been facilitated in the definition by including a set of lower level behaviours. On the lowest level the behaviour is linked to a physical phenomenon \(<PP>\) that describes it. The physical phenomena are described using informal descriptions of the physical principle and a formal description of the types of valid objects (input) and how they change (output).

\[ < PP >_i = \{ description,(O_{j},...,O_{k})_{in},(O_{n},...,O_{m})_{out} \} \] (9)

ILLUSTRATIVE EXAMPLE

To illustrate the introduced function-behaviour-structure model and to discuss its merits for the definition of modular assembly system equipment a simple example has been chosen. Since the overall aim is to provide a model that will aid the configuration process of modular components into systems, we will start with the definition of a required assembly process. The “peg in hole” example has been picked for illustrative purposes.

The first step of understanding which module will be required is to break down the overall assembly process into its constituting functions and precedence constraints. Figure 5 shows a possible sequence of functions to insert a round peg (part A) into the corresponding hole (in part B). It has been assumed that the two parts are in defined locations and that the position of part B coincides with the assembly position. All the functions shown can be abstracted as an \(<insertion>\) function. The complete assembly process definition of these two parts would require additional functions that define the system requirements for feeding, storing
continuation after the assembly and potential handling operations in the form of, for example, “avoid obstacle x”.

For this example process, the function types used are <hold>, <release> and <move>. All three of them can be linked to combinations of translations and or rotations. Furthermore, <release> is the inverse function of <hold> and could therefore be embodied with the same device. Both <release> and <hold> need at least one translation against a fixed point or two opposite translations. This requirement fits the gripper in Figure 4. The positions of the <move> functions indicate that at least a two DoF device will be needed to embody it. Figure 6 and Figure 7 show two different devices that both have more than two DoF and therefore fulfil the function requirement. The last question remaining is whether these two devices can actually be connected to fulfil the whole required process. This is the case as long as their interfaces are compatible.

**Figure 6 Structure and behaviour of a simple pneumatic manipulator**

**Figure 7 Structure and behaviour of a SCARA-type robot**

**CONCLUSIONS**

This paper has defined an ontology based data model for the definition and configuration of assembly equipment modules based on their functions, behaviour and structure. The model has been demonstrated with an illustrative example that shows the strength of the model for
computer driven synthesis of modular assembly systems. Particularly the use of physical phenomena to objectively describe the capabilities of a module in conjunction with ontological definition of the data provides a very powerful tool to enable computational methods to aid the design process.

Future work will be focused on the exploration of this model in different design scenarios as well as the formalisation of synthesis methods for conceptual design of modular assembly systems. The key challenge in this area is perceived to be the integration of artificial intelligence approaches to define the synthesis process in an adaptive manner.

ACKNOWLEDGEMENTS
The reported work is partially funded by the Department of Trade and Industry in the United Kingdom as part of the EUREKA Factory E!2851 E-RACE project the support of which is gratefully acknowledged.

REFERENCES