Essential DCI: Data Context Interaction

Goals

DCI is a paradigm used in computer software to program systems of communicating objects. Its goals are:

- To improve the readability of object-oriented code by giving system behavior first-class status;
- To cleanly separate code for rapidly changing system behavior (what the system *does*) from that for slowly changing domain knowledge (what the system *is*), instead of combining both in one class interface;
- To help programmers reason about system-level state and behavior instead of only object state and behavior;
- To support an object style of thinking that is close to peoples' mental models, rather than the class style of thinking that overshadowed object thinking early in the history of object-oriented programming languages.

Constraints from the DCI Execution Model

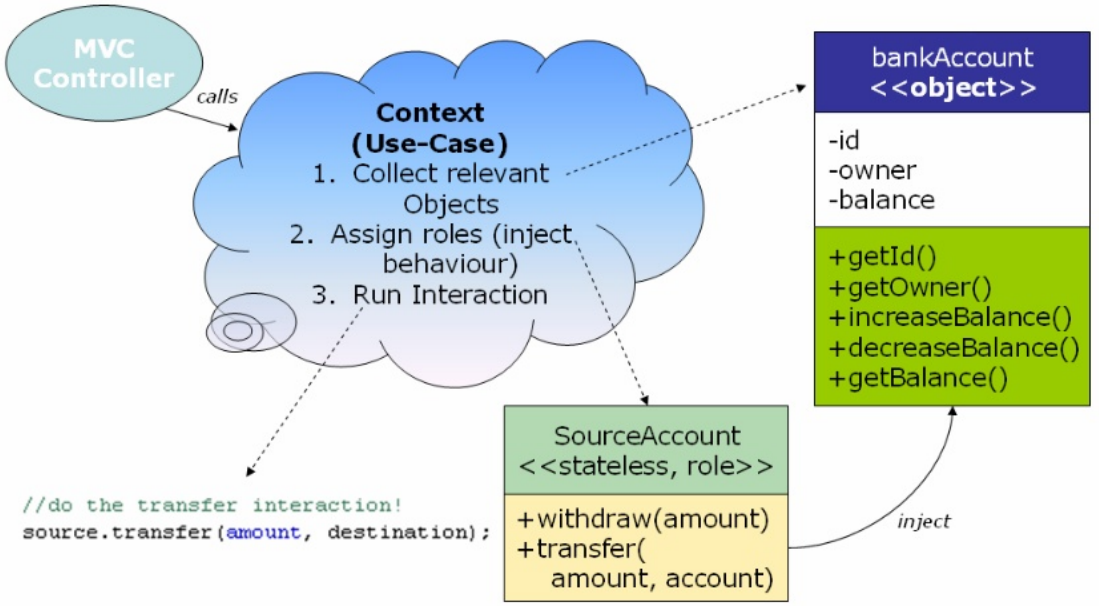
- *All Roles in a Context are bound to objects in a single, atomic operation.*
- *The name of a RoleMethod must be unique within the Role and also within all objects that may play this Role.*
- *Only one DCI Context can be active at a time.*

with DCI, the application facade that used to look something like this:

facade.method(id1,id2,id3,param1,param2)

is now replaced by:

context<id1,id2,id3>.interaction(param1,param2)

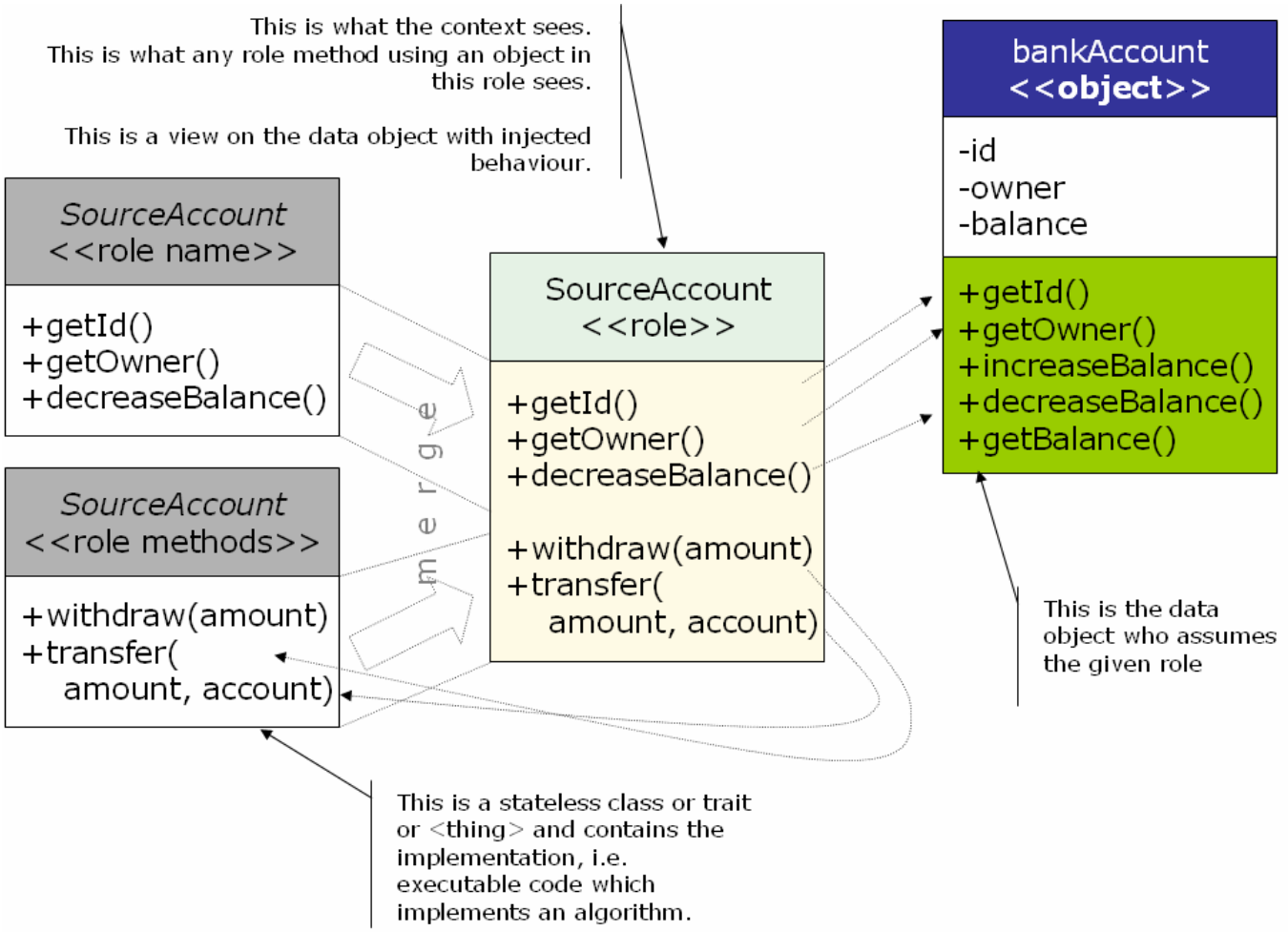


DCI was invented by Trygve Reenskaug, also the inventor of MVC. The current formulation of DCI is mostly the work of Reenskaug and James O. Coplien. Public awareness increased after a session at the JaOO conference in Denmark in September 2008.

Useful Links

http://architects.dzone.com/videos/dci-architecture-oberg
http://en.wikipedia.org/wiki/Data,_Context,_and_Interaction
http://heim.ifi.uio.no/~trygver/2010/DCIExecutionModel.pdf
http://heim.ifi.uio.no/~trygver/2009/commonsense.pdf
http://groups.google.com/group/object-composition

Essential DCI: Data Context Interaction



DCI makes reviewing / reading / understanding code easier by...

...formally separating the Data Model (what-the-system-is) from System Behaviour (what-the-system-does);
...narrowing each object involved in an interaction (by using a role-contract), so that anyone reading the code only needs to know about that narrowed interface;
...formally relating implementation directly to use-cases;
...relating unit tests, directly to use-cases because when unit tests are written for contexts, they automatically test use-cases, rather than unit testing random parts of use-cases;
...making it very clear where you need to invest your testing effort, i.e. unit test the roles.

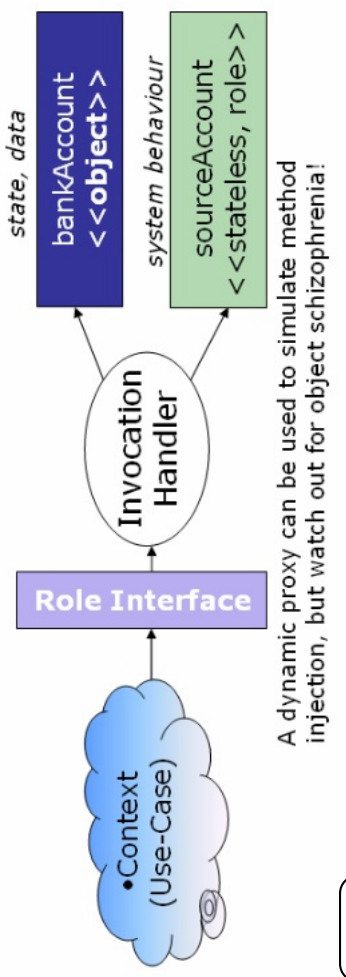
- Roles can only exist within a context. Roles are not useable outside of the context for which they were written, and as such can be implemented within the context class.
- A context is responsible for assigning roles to data objects, and starting the interaction by calling a role method on one of the role players.
- A context does not usually contain algorithms relating to system behaviour, rather roles do.
- A context in a statically typed language may not contain references to data objects if it is to be usable in the future by data objects currently not conceived by the context & role author.
- Hence, a context should only use data objects once they are playing roles.

Essential DCI: Data Context Interaction

DCI...

...reduces the size of the inheritance tree of the data model, when compared to OOP;
...makes reading the code sound more like the users mental model, when compared to a services solution;
...can be combined with frameworks or application servers which consider cross-cutting concerns like transactions and security, as well as other concerns like resource management, concurrency, scalability, robustness and reliability – it is the context which the container knows and deals with.

- *Data... is what is operated on and changed and it is system state.*
- *Roles... are the behaviour which objects can take on for a given use case or context.*
- *Contexts... are use cases and collect data which needs to interact within a use case and casts it into roles capable of carrying out the use cases. The context starts the interaction.*
- *Interactions... are conceptual and are the action of a context making a call to a role player in order to do work. They can also be the individual methods which a context exposes to its caller.*



Java Example

```
// prepare the injector (dynamic proxy)
BehaviourInjector behaviourInjector = new BehaviourInjector(this);

// convert the domain object into a role, and inject the relevant
// role methods into it
ITransferringSourceAccount_Role source = behaviourInjector.assignRole(
    sourceAccount, //domain object
    ITransferringSourceAccount_Role.class); //the role interface
```

Scala Example

```
def withdraw(accountId: Int) {

    var source = new BankAccount with SourceAccount_Role

    source.withdraw(100)

    println(source getBalance)
}
```

SELF: the object, currently playing the role being executed

```
//decrease balance
self.decreaseBalance(amount);
```

More Useful Links

http://vimeo.com/8235394
http://vimeo.com/8235574
http://www.artima.com/articles/dci_vision.html
http://www.maxant.co.uk/whitepapers.jsp

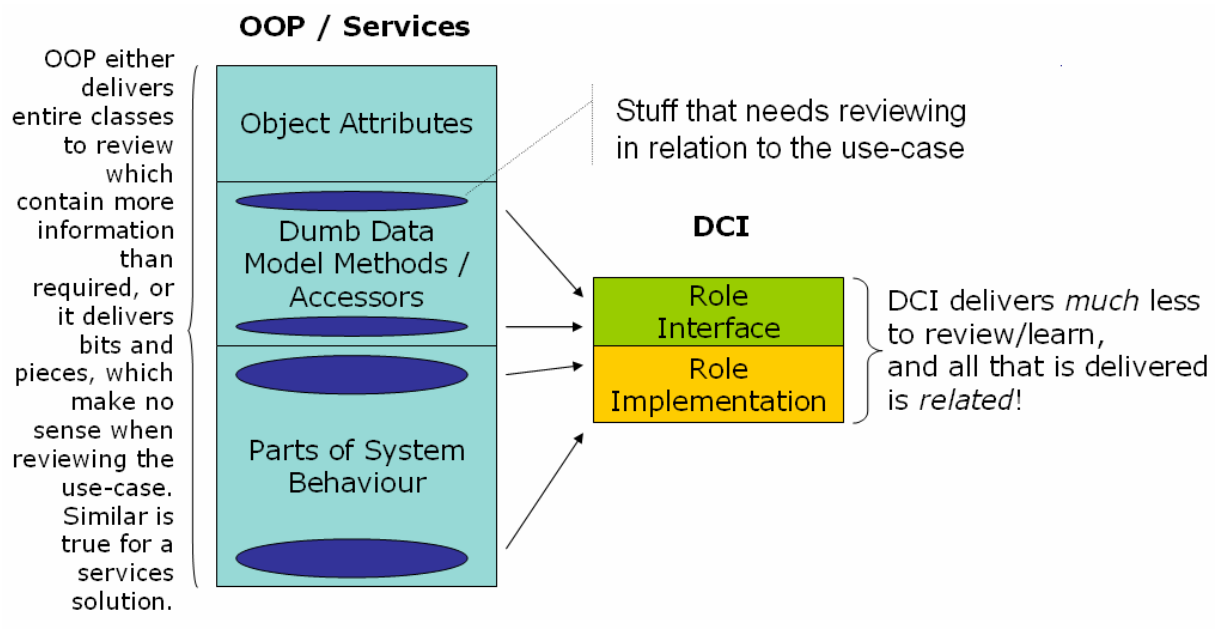
Essential **DCI**: **D**ata **C**ontext **I**nteraction

- RoleMethods are the methods containing implementation. The are located in Methodful Roles.
- RoleNames are identifiers used in RoleMethods. Calling a method on a RoleName calls a method located in either the RoleMethod, or the Data Object. The interface which a RoleName exposes is a mix of RoleMethods and Data Methods. Normally, this interface only includes the data methods used in all role methods, rather than all methods in a particular data objects. This allows roles to be assigned to objects more generally.
- Methodless Roles are RoleNames.
- Methodful Role are traits/classes/mixins/things which contain RoleMethods, ie. the place where RoleMethods are implemented.
- A RoleContract is the interface a role requires a data object to have in order to play the role.
- A RoleInterface is the interface which a data object playing a role exposes, ie. the RoleContract + RoleMethods
- A RolePlayer is an object playing a role
- A RoleMap is a mapping inside the context relating the role name to the role player
- Data objects are *assigned roles*, or can be *cast into roles*, although the former is preferred.

Hamlet Analogy by Rune Funch Søltoft: A playright (the User) writes a script (the Use-Case). As part of the script, roles are created which make actors (Data) Interact. The casting director (the Context) casts the best suited actor for each role. If no suitable actor is found, you do not re-write the script, rather you re-cast the role to a more suitable actor. The use-case defines data and interactions. These interaction are used to defined roles. As such, the actions described in use-cases become role methods and the entities become the data model.

DCI applies to object oriented programming and is a paradigm for improving object oriented programming. Its focus is to allow objects to be enriched with functionality at the time when they need it, rather than them having all the functionality they may ever need from the start.

The Context is the class (or its instance) whose code includes the Roles for a given algorithm, scenario, or use case, as well as the code to map these Roles into objects at run time and to enact the use case.

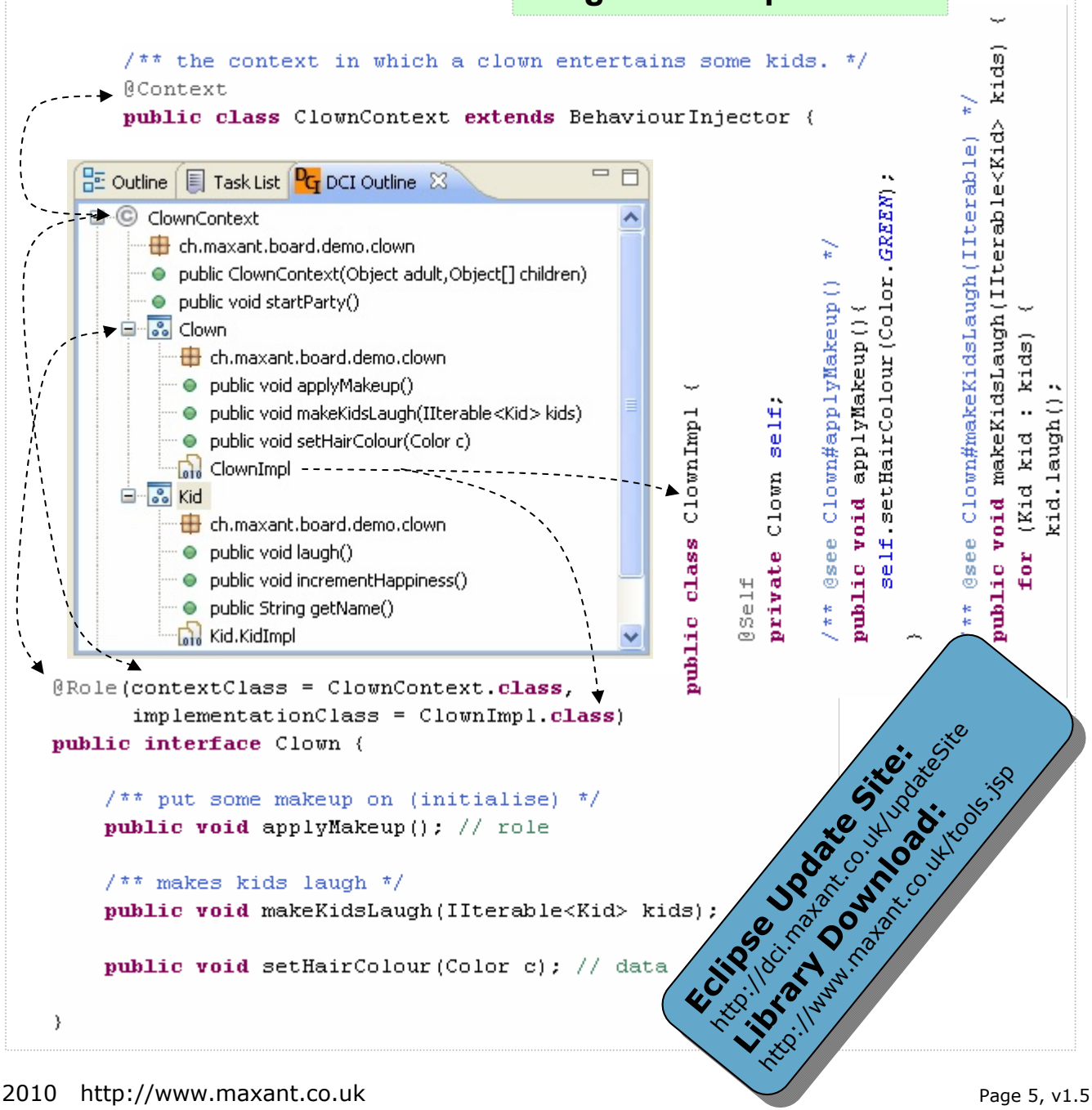


Essential **DCI**: Data **C**ontext **I**nteraction

DCI architectures tend to be characterized by the following properties:

- The Data model reflects the domain structure rather than partitions of its behavior;
- Objects dynamically take on Roles during use case enactments;
- Every Role of a use case is played by an object determined by the Context at the start of the use case enactment;
- The network of Interactions between Roles in the code (i.e., at coding time) is the same as the corresponding network of objects at run time;
- These networks are potentially re-created on every use case enactment;
- Roles come in and out of scope with use case lifetimes, but objects that may play these Roles may persist across multiple use case lifetimes and may potentially play many Roles over their own lifetime.

Plugin for Eclipse / Java



Eclipse Update Site:
http://dci.maxant.co.uk/updateSite
Library Download:
http://www.maxant.co.uk/tools.jsp