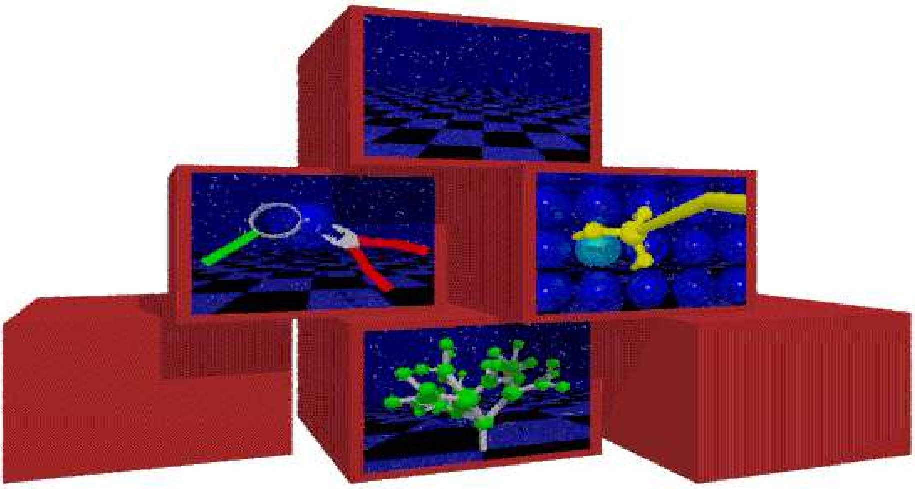
CAP

The CyberSpace Architecture Project

The CyberSpace Foundation

More than a Class Library

*This paper gives an overview of the CyberSpace Foundation (*CSF*), the base of the CyberSpace Architecture. It is shown how various software aspects are captured, how access to them is provided and which advantages follow from the abstract and general concept of the* CSF. *The main categories – Space, Dialog, Access and Structure – are presented and the interaction with the CyberSpace Object Architecture (*COA*) and Object Oriented Views (*OOV*), two further concepts of* CAP, *is explained.*



The CyberSpace Foundation provides basic building blocks.
Main categories are Space, Dialog, Access and Structure.

1 Introduction

The CyberSpace Foundation is the base of CAP. It provides a set of fundamental terms for the design of software systems in form of highly abstract base classes.

The CSF captures a wide range of aspects of software systems such as data structure (bags, sets, lists etc.), presentation and manipulation (user interfaces), persistency (data bases), spatial aspects, location etc.

The design of CSF does not focus and is not bound to any specific implementation. Instead the basic properties of the "CyberSpace" itself are examined. This leads to a high degree of generality and therefore portability of the CSF classes.

This basic CSF interface allows to encapsulate several specific services like more technical standard class libraries, user interfaces, data bases, communication libraries etc.

The CSF serves as an anchor for these services. The access to more specific features is of course possible by using derivation and other object oriented features.

2 Properties of the CyberSpace

What are "properties of the CyberSpace", and how are they related to everyday's work with computers?

The term CyberSpace is used within CAP to denote the complete world of software. This is why it is related to everyday's work with computers. There is surely a difference between a virtual reality spacecraft journey and working with a spreadsheet, but visualisation aspects are not primarily meant here[1].

Hence examining the properties of CyberSpace means to search for common properties of software of all kinds. Surprisingly, there are in fact such properties, and some of them are known since software is known.

The CyberSpace Foundation tries to capture these basic properties in terms and to provide them as "software terms": as abstract base classes.

As far as they have been identified these classes can be classified into the following main categories:

Space
Properties of the space itself without any content are the first category. These properties are important as the space can serve as a valuable reference system, like the physical space, which provides coordinates for locations and can be devided into subspaces for ordering the world.

Dialog
To act in the space relies on exchanging data between objects (or subjects). This is the only possibility to effect something in the world of data, therefore in analogy to the physical space message exchange may be compared to forces. This category captures mainly the dialog between objects and human beings, but also dialog between objects and objects.

Access
To act on a data object, it is necessary to have some kind of access to it. Depending on the kind of operation, different kinds of access are appropriate. Searching for objects, identifying them or having the possibility to access them are problems belonging into this category.

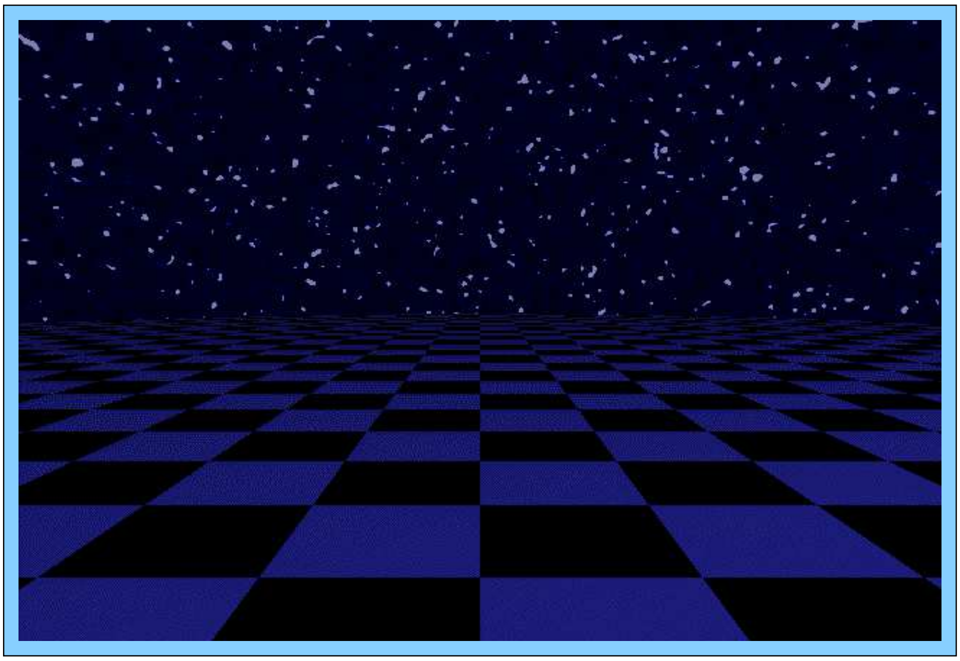
Structure
Last not least, structure is an important property of things. Data structures are explicitly known and have been discovered in the early days of computing. There are already several implementations of class libraries available handling this aspect on different levels of abstraction.

These main categories are explained in more detail in the following sections.

[1] The meaning of Cyberspace used here is in fact the meaning which was originally introduced in William Gibsons science fiction "Neuromancer".

3 Components of CSF

3.1 Space



The *Space Classes* of the CyberSpace Foundation capture properties of the space itself. The Space serves as a valuable reference system, like the physical space in the real world. The space defines locations.

This reference system can be used to bind various kinds of information to, to define areas associated with physical, geographical, administrative or even arbitrary regions. Furthermore, it may be used to express a wide range of operations on a logical level, like moving, copying or linking things.

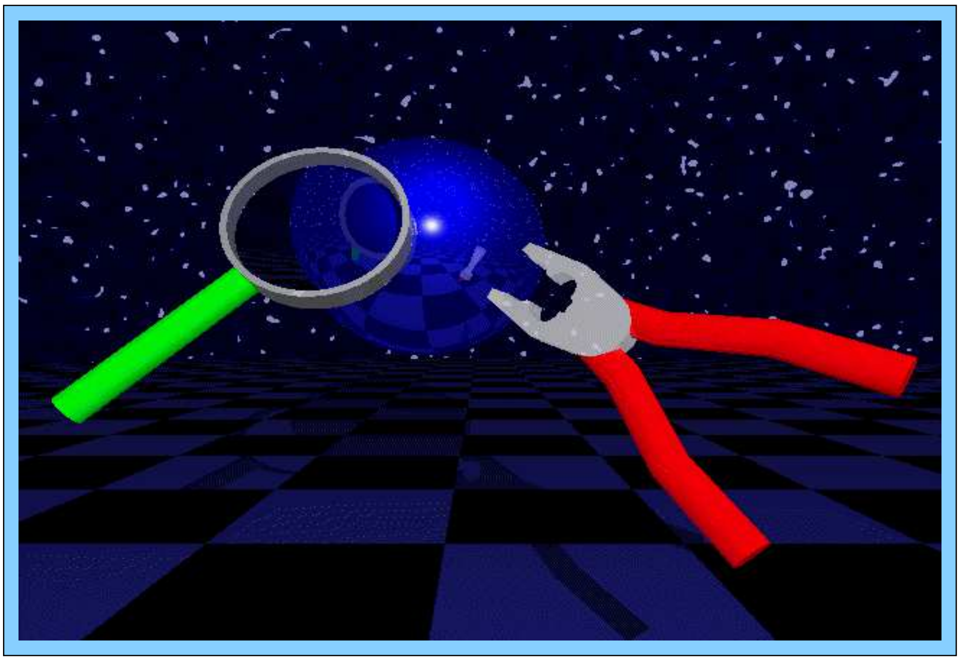
Contextual information can be bound to areas and provided explicitly. This allows to perform automatically operations like installation or conversion of formats or languages.

Persistency may be viewed as an attribute of a region. If it is at all necessary to model this property on abstract levels, the space provides a good frame.

Problems related to the identity or distribution of objects and copies of them can be handled easier if spatial aspects are considered, since even in CyberSpace no two objects can share at the same time the same place, therefore identity is naturally preserved.

For further details on the Space Classes of the CSF see [?].

3.2 Dialog



The *Dialog Classes* capture the interaction between actors, mainly human beings, and objects to operate on.

This includes the presentation of objects in every imaginable form, and the provision of the information which is necessary to create such presentations. It also includes the provision of manipulators, which allow to operate on objects on many different appropriate levels of abstraction.

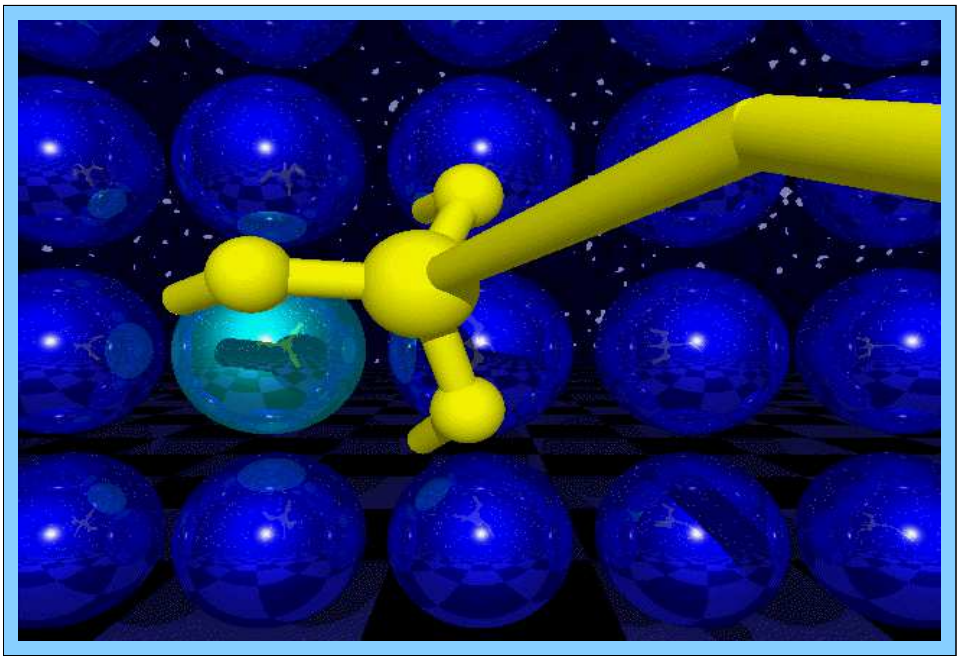
The Dialog Classes provide this presentation and manipulation data in a fine grained, generic form, which allows the dynamic and automatic construction of user interfaces of any kind. This supports not only portability and reusability of interface data, but features like *dynamic runtime* portability and persistency of interfaces and their states on a surface independent level!

Due to the very general and abstract approach of the Dialog Classes they are really easily extendable to new interface technolgies, like e.g. virtual reality.

Objects, in contrast to human beings, as actors behave very different and have different requirements (most likely more simple ones), which are handled also to some extent by classes of the dialog category.

For further details on the Dialog Classes of the CSF see [**?**].

3.3 Access



Access is the precondition to be able to act on something. The *Access Classes* capture various aspects of handling access to objects.

Searching is the first step to have access. There are classes provided to specify search criteria, i.e. features, constraints, types etc. in a general form.

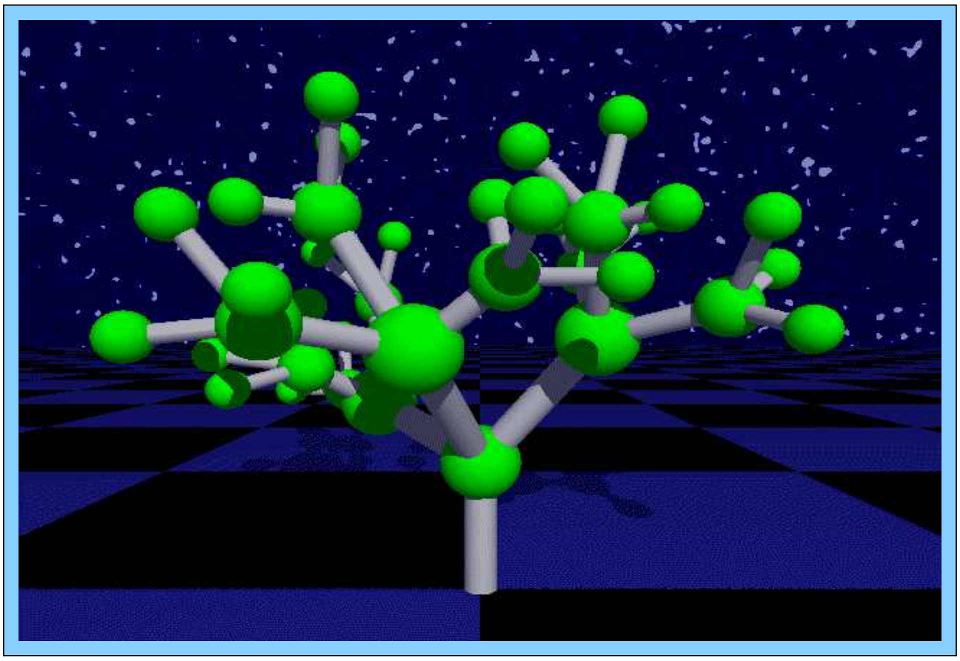
Related to searching is the problem of identifying objects. Search criteria may be distinguished from identifying features by using different classes to model them.

If an object is found, but access is currently not necessary, the use of *Pointers*[2] is most appropriate, to express the preservation of the possibility of access. A pointer is also useful if the identity of the object is not important, but the existence of an object of the appropriate type.

For further details on the Access Classes of the CSF see [**?**].

[2] No, the term pointer does not denote an implementation construct in this context.

3.4 Structure



Last not least, the *Structure Classes* handle an old known important property of data. Lots of class libraries already exist which handle these aspects on variuos levels of abstraction.

The Structure Classes of the CSF shall not represent the 1000th implementation of a data structure library. But since there is no standard yet the CSF shall be completed with such classes.

Furthermore, the Structure Classes are probably a bit more abstract than the most libraries existing and a bit more complete, in that it tries to integrate "Sets", "Relations" etc. in a coherent manner.

And finally, there is in fact not something like "the one and only" modelling of structural aspects. But the CSF Structure Classes are prepared for interaction with the CyberSpace Object Architecture, which allows the coexistence of different models and to use the respective appropriate one.

For further details on the Structure Classes of the CSF see [**?**].

4 CSF, COA and OOV

The relation between the CyberSpace Foundation and the CyberSpace Object Architecture (COA) is simply, that CSF is the content and COA provides the form and access.

From another point of view, COA is an important addition to CSF.

Since no abstraction is really finally abstract but is more or less appropriate to a specific kind of use, and due to the openness of CSF their must be the possibility for coexistence of and cooperation with further abstractions.

These features are provided by COA, which therefore represents the openness, while CSF represents some kind of preconfigured content.

The relation to Object Oriented Views (OOV) is similar. CSF objects are used as components of OOVs, which provide the frame. The CSF classes may serve as valuable anchors to build small class hierarchies for usage wihtin OOVs.

5 Conclusion

The CyberSpace Foundation captures main categories of the CyberSpace, the world of software and data.

Besides well known categories like data structure and data access, the concept of the space is introduced. Furthermore, user interfaces are elaborated to a much wider extent.

Alltogether, the CSF classes provide a powerful and well-rounded set of basic abstractions.