Product Model Based Software for Structural Design

Miikka **Kiiski,** Department Manager
Tekla Oy
Espoo, Finland

Abstract

Tekla Oy is developing a software package called Xbuild for the design and detailing of steel and concrete structures. The development focuses on two main areas: steel structure design and design of reinforced concrete structures. Accordingly, Xbuild consists of two main parts: Xsteel and Xconcrete software modules.

The basic idea behind the Xbuild is to build a logical product model of the steel/concrete structure. This product model is stored in a relational database and it is created by using sophisticated interactive 3D-modelling tools. All documentation needed for the manufacture and construction of the structure - drawings, material lists, NC-preprocessor files - can then be produced from the product model.

Xsteel includes modelling tools for beams, columns, connections, plates, weldings, bolts and other components of a steel structure. Most of the standard components used in Finland and other European countries are stored in component libraries such as profile, connection and bolt libraries. The modelling is object-oriented, which makes the model "intelligent". Every component in the structure is an object in the product model database and objects can be connected to each other by certain rules. In practise this means that for instance when a beam is being moved, the adjoining joints will follow. Every object is stored in the database only once, which ensures the coherency of the database in all situations. The 3D-model, drawings and lists are just "views" to the database - all design modifications can only be made in the model. This way the user can be sure that all documentation of the model is always up to date.

Xconcrete is based on the same principles as Xsteel. The main difference is that Xconcrete can also handle the reinforcement bars in an intelligent way by utilizing object-oriented techniques.

The database structure of Xsteel and Xconcrete is relational. The contents of the database can be written out in any format specified by the user. This enables data transfer between Xbuild and any other product model based software. It is also possible to link other applications, such as strength analysis and dimensioning, production planning and cost calculation, to Xsteel by using an open linking interface. In addition to this, the Xbuild software modules include tools for creating

Construction Informatics Digital Library http://itc.scix.net/
paper w78-1994-37.content

user specified macros - a feature that enables users to develop own Xbuild "applications".

Today Xsteel is used by several engineering and steelwork companies in Finland as well as abroad. Xconcrete is still partially under development and will be completed in the near future.

As the construction process, codes of praxis etc. differ a lot in different countries, the requirements set on the software vary quite much from one country to another. Therefore the software has to be easy to adapt into different desing environments.

The results gained by the users show that the product model based approach is radically improving the productivity and quality of the design work. On the other hand it is clear that using a sophisticated product model based design software sets new requirements for the designers and manufacturers of structures.

1 Xbuild SOFTWARE FOR STRUCTURAL DESIGN

Xbuild is a product model based software package for the design and detailing of steel and reinforced concrete structures. Xbuild consists today of two software modules: Xsteel and Xconcrete. Modules for cost calculation and statical analysis (FEA) will be added or linked to the Xbuild package in the future.

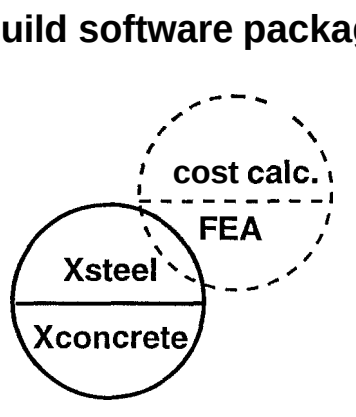

Figure 1. The Xbuild software package

1.1 PRODUCT MODELS

The basic idea behind all the Xbuild software modules is to build a logical product model of the structure. This product model includes information of all the objects (elements) of the structure. The product model is hierarchical and there is a set of

rules that connects the various objects to each other. All objects include a set of attributes that define the charasterics of that object. Figure 2. shows an example of an object (endplate) that is connected to another object on an higher level of hierarchy (endplate connection) in a steel structure.

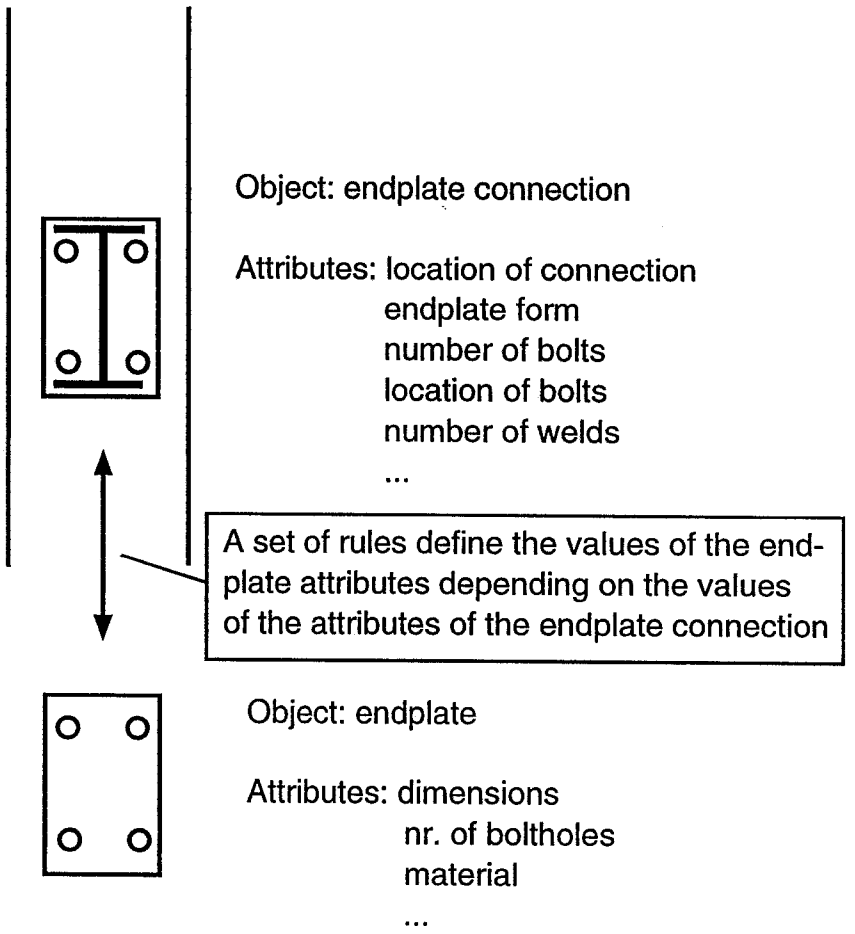

Figure 2. Two objects of a steel structure.

The product model is created by using object-oriented programming techniques, which enables the efficient use of object hierarchy and logical connections and rules between objects. In practice this means that, for instance, if the size of one of the connected beams in figure 2. would be changed, the connection attribute "endplate form" and the endplate attribute "dimensions" would be changed accordingly.

1.2 SOFTWARE ARCHITECTURE

The Xbuild software has been developed by using Tekla Oy's "kit" family of software development tools. These tools enable fast development cycles of application software and guarantee a high quality in all software modules. Figure 3. shows the software architecture solution.

1.2.1 Database, virtual database and DB-kit

The product model database is based on a relational database model. All information about the model, including both the geometrical representation and the attributes of the objects, is stored in this database.

When starting a Xsteel/Xconcrete session, the model database is read into a "virtual database" that runs in the RAM of the computer. DB-kit is the tool that handles this virtual database and enables extremely fast add-, delete-, modify- and other logical operations.

1.2.2 Graphics, X-kit

The graphical on-screen representation of the model is done by using X Window graphics. The graphical 3D-representation of the model is actually a "view" to the model database. The X-kit tool is utilized when creating the on-screen graphics.

1.2.3 Graphical user interface, A-kit

The GUI is based strictly on OSF/Motif standard - Windows NT will also be available in the future. The A-kit (application kit) tool is used for the development of the graphical user interface.

1.3 HARDWARE REQUIREMENTS

Xbuild software runs today on Unix workstations using X11 graphics library and OSF/Motif user interface. The recommended workstation configuration includes 32MB RAM, 300MB available disk space and a 19" color screen.

2. Xsteel SOFTWARE FOR THE DESIGN AND DETAILING OF STEEL STRUCTURES

Xsteel is a software module for the design and detailing of steel structures. Xsteel is used by several consulting engineers and steelwork companies in Finland as well as in several other European countries.

2.1 Xsteel PRODUCT MODEL

The structure of the Xsteel product model is shown in table 1.

Hierarchy Level	Object types
1.	Model
2.	System (sub-model)
3	Higher level objects (groups of basic elements such as stairs and railings)
4.	Basic objects: Beams, columns, contour plates, connections,details (footplates, stiffeners, fittings)
5.	Lower level objects: Boltgroups, weldings, cuts, holes, plates

Table 1. Xsteel product model.

The user creates a model that can be divided to several systems (submodels). Normally the modelling is done by creating the basic objects and defining their attributes and locations. The basic objects can also be groupped to form higher level objects like stairs and railings. The basic objects may consist of several lower level objects that are connected to the "mother object" by a specified set of rules. Also the basic objects can be connected to each other by certain rules.

The use of rules makes the product model "intelligent" - for instance if one beam in the model is moved to another location, all connections and other beams that are connected to it will chance accordingly.

2.2 ATTRIBUTES OF BASIC Xsteel OBJECT "BEAM"

This is an example of the attributes of a basic Xsteel object. The object "beam" contains attributes that define its location in the model (global coordinate system), rotation and offsets of beam ends (local coordinate system), identifiers (position number, assembly number) and profile type and material data.

Table 2. shows all beam attributes, their meaning and an example of the values of an attribute.

Attribute	Use	Example value
Part. nr	Unique part nr. of a beam	P-101
Assembly nr.	Nr. of an assembly where this beam is used	A-2
Name	Object name	BEAM
Profile	Profile type	HE300A
Material	Material	FE360B
Class	Material class	1
Radius	Radius of curved beam	0,0
Move end 1 (Dx, Dy, Dz)	Offset of end 1	150.00 (mm)
Move end 2 (Dx, Dy, Dz)	Offset of end 2	0.00 (mm)
Position in plane	Position of beam in workplane	Middle
Rotation	Rotation of beam (local coordinates)	Top
Position in depth	Postion of beam in/out of workplane	Behind

Table 2. Beam attributes

2.3 CREATION OF PRODUCT MODEL (MODELLING)

The Xsteel product model is created by using an interactive three dimensional modeller. This modeller provides the user with basic "CAD functionality" to move around in the model, to define locations of objects and to modify and delete existing objects. The object attributes are given by using "dialoque boxes" that are based on standard OSF/Motif windows.

The user can have as many views to the model as needed, each view is placed in a separate window. These views can basically show the model from any direction and the user can also zoom in and out the model freely. Figure 4. shows an exam-

ple of an orthogonal view of an Xsteel model.

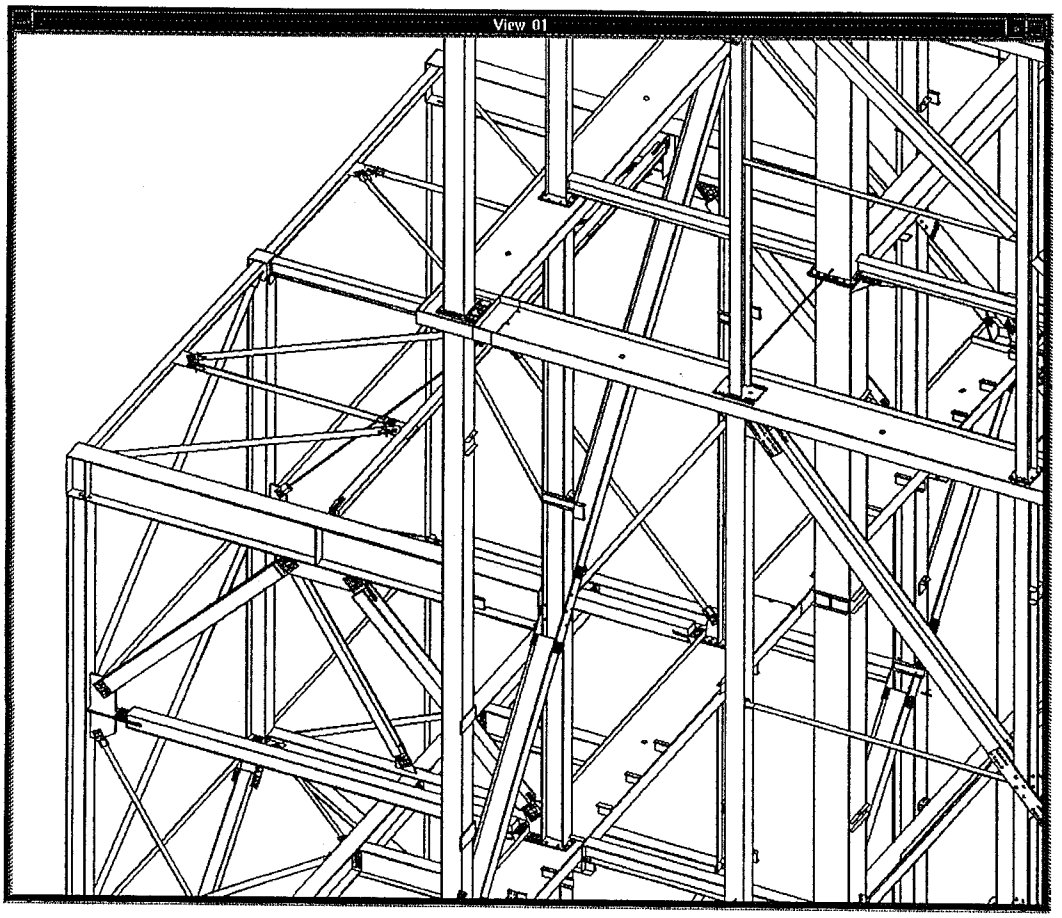

Figure 3. An orthogonal view of an Xsteel model.

2.4 STANDARD OBJECT LIBRARIES

The Xsteel user is provided with standard object libraries such as the profile/cross section libraries (DIN and U.K.) and bolt libraries (SFS, DIN, U.K.). The user can also modify these existing standard object libraries and create new ones when needed.

2.5 OUTPUT FROM THE Xsteel PRODUCT MODEL

The output generated from the Xsteel product model can be divided into graphical output (drawings) and text output (lists and files). This output is used for several purposes during the design, manufacturing and building stages of the structure.

2.5.1 Drawings

The drawings generated from the Xsteel product model can be basically divided into three cathegories: general, assembly and workshop drawings.

The general drawings are being used throughout the design, manufacturing and building stages of the structure to give an idea of the whole structure or certain parts of it. These drawings can represent for instance the whole structure in an isometric view or certain floors and sections of the structure in orthogonal views.

The assembly drawings typically show one assembly and give instructions of how the different parts should be connected together. Figure 5. shows a typical assembly drawing.

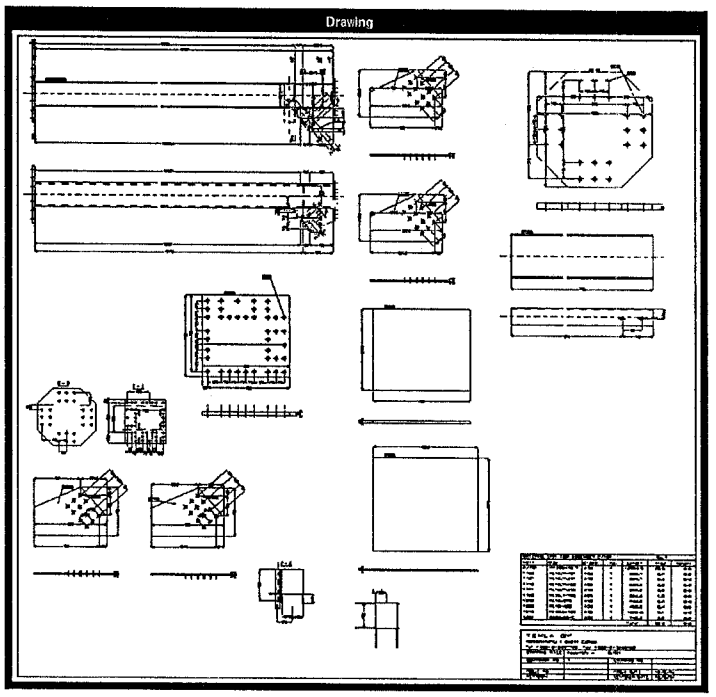

Figure 4. Assembly drawing.

The workshop drawings typically represent one single part of the structure and give instructions of how these parts should be manufactured.
All the drawings are tightly connected to the product model database. If there are changes in the product model, the software shows which drawings are effected by these changes and the user can automatically update all the altered drawings.

Changes in the product model can only be made by using the 3D-modeller, which ensures the coherency of the product model database at all times.

2.5.2 Lists and files

Several types of lists, including part lists, bolt lists, profile lists etc. can be generated from the product model database. Xsteel provides the user with the basic tools (report generator) to define the contents and the format of the lists, but also a selection of predefined list formats is given to the user.

Another form of text output from the product model database are various output files. These files enable data transfer from the product model to other applications such as preprocessors for numerically controlled tooling machines, production planning applications and other product model based design systems.

3. Xconcrete SOFTWARE FOR THE DESIGN AND DETAILING OF REINFORCED CONCRETE STRUCTURES

Xconcrete is a software module for the design and detailing of reinforced concrete structures. The basic ideas behind Xconcrete are very close to those of Xsteel, so mainly the features of Xconcrete that differ from Xsteel will be discussed here.

3.2 Xconcrete PRODUCT MODEL

The structure of the Xconcrete product model is shown in table 3. The structure of the product model is basically similar to the Xsteel product model. The main difference is the ability to handle the reinforcements in concrete elements.

The importance of rules that define the relations between objects is even greater in Xconcrete than in Xsteel. This is due to the fact that these rules define, among other things, the connection between a conrcete element and the reinforcement inside it. Also the way single reinforcement bars are located in a reinforcement is defined by these rules.

Hierarchy Level	Object types
1.	Model
2.	System (sub-model)
3	Higher level objects (groups of basic elements such as supplies)
4.	Basic objects: beams, columns, slabs, walls, connections, details, reinforcements
5.	Lower level objects: reinforcement bars, stirrups, cuts, holes, plates

Table 3. Xconcrete product model

3.3 MODELLING OF REINFORCEMENTS

The modelling of reinforcements is typically done by either modelling single bars or groups of bars that form nets and reinforcements. The reinforcements are connected to the concrete elements by a set of rules that define the relations between these objects. Also the reinforcement attributes are used to define the location of reinforcements inside concrete elements. These attributes include bar dimensions, concrete coverage, bending radius, anchor length, steel quality and bar type definitions.

The reinforcements can be manipulated by changing the attribute values or by using similar manipulation commands as with concrete elements. A single bar or a group of bars can also be manipulated by moving or deleting bar vertexes or by adding new vertexes and thus changing the shape of the bars.

The reinforcements are normally modelled by using reinforcement macros. These macros can also be written by the user.

3.4 STANDARD OBJECT LIBRARIES

The standard object libraries of Xconcrete include profile/cross-section libraries, reinforcement libraries and libraries of connections and supplies. These libraries can also be modified by the user.

3.5 OUTPUT FROM THE Xconcrete PRODUCT MODEL

The principles of the output from the Xconcrete product model are the same as in Xsteel. The output can graphical (drawings) or text (lists and files). Today there are some Xconcrete applications that produce the pre-processor data for numerically controlled reinforcement bar cutting and bending machines.

4. LINKS TO OTHER SYSTEMS

The structure of the Xbuild product model database makes it possible to create links from the Xbuild software modules to other applications. These can include desing and statical analysis software, other CAD applications, production planning and CAM applications.

The links between Xbuild and design/statical analysis applications are typically done by exporting the geometry of the product model from Xbuild, analysing the structure in an outside application and importing profile and cross section data back to Xsteel.

The links between Xbuild and other CAD applications can be created by exchancing either 2D drawings or product models between the systems. Drawings created by Xsteel can be converted into dxf format, which most of the 2D CAD systems like AutoCAD or MicroStation can read. The product models can be transferred by exporting/importing them in a specified ASCII text format. Such links have been made from Xbuild to Intergraph's Modeldraft steelwork desing system and CadCentre's PDMS plant desing system among others.

The links to CIM applications (production planning, CAM) are typically created by using ASCII text files that contain the needed information from the product model.

5. Xbuild MACRO LANGUAGE

In both existing Xbuild applications (Xsteel and Xconcrete) the user is provided with a macro language that enable the development of user-defined macros. These macros can be used to create new connection types, higher level objects, new reinforcement types etc.

The Xbuild macro language is a set of subroutine calls written in ANSI C language. These call for the Xbuild subroutines that are being used to create the objects in the Xbuild product model. In the basic Xbuild are included libraries of macros (connections, reinforcements etc.) and by modifying these macros, or writing completely new ones, the Xbuild users can easily write new "applications" that

match to their needs. In Appendix 1. is shown a typical connection macro. The Xbuild macro language has proven to be a very efficient tool giving the end-users the freedom to modify the software to match their special needs.

6. Xbuild TODAY AND IN THE FUTURE

6.1 Xsteel

The Xsteel software is used today by several consulting engineering and steelwork companies in Finland and other European countries. The market for product model based steelwork design and detailing software is growing steadily. In most European countries the idea of product modelling as an alternative to the traditional, drawing based CAD approach, seems to have gained general acceptance. Pan-European development projects, such as CIMSteel (Eureka Project 130), have obviously been of great help with promoting the new ideas among the industry.

Xsteel development continues and new versions of the software are being released twice a year. Today the development is mainly focused in localising the software to match the needs of the various target markets. Creating closer links to other CAD systems and other outside applications is also one of the key issues in future Xsteel development.

6.2 Xconcrete

The Xconcrete software is still partially under development. There are some cases, where parts of the software are being used for production purposes, but as a whole, the complete application will not be availale to end users until 1995. Today the Xconcrete development is mainly focusing in things like creating the standard object libraries, defining the output formats, adding new functionality and macros to the core software etc.

6.3 NEW Xbuild SOFTWARE MODULES

One of the new Xbuild development projects is called "Xcalc". This is a codename for an application that can be used to calculate the costs of a structure that has been modelled by Xsteel or Xconcrete. The Xbuild product model is naturally an ideal basis for cost calculations, as all the information about quantities and materials is already there. Adding information about the costs of labor and material and defining the rules between the objects of the product model and these cost is needed to produce accurate and reliable cost estimates.

6.4 MARKET-SPECIFIC REQUIREMENTS

The requirements set on structural design software vary quite much in different markets and countries. This is partly due to different standards and codes of praxis in different countries, but also the way desing work is organised plays an important role here. In some countries it is typical that structural design and detailing is mainly done by consulting engineers, who produce drawings and material lists for the workshops. In other countries the consulting engineers produce only the general plans and drawings and the detailing is done in the workshops or specialised detailing companies. Sometimes the workshop does everything: design, detailing and manufacturing.

The special requirements set on 3D modelling concern typically the profile and reinforcement bar libraries, connection macros and other structural details. The open structure of the standard libraries, together with the macro language, makes it normally quite easy to modify the software to match these special needs.

In drawing production the special requirements are typically set on the outlook of the drawings. The drawing sizes, drawing frames, rules and symbols of dimensioning and other details are normally things that need customising from country to country. Most of these requirements can be met by modifying the various drawing attributes of the Xbuild software modules.

The lists and reports are normally modified by each user to match their specific needs. The Xbuild software modules include specific tools that can be used to modify the contents and format of the lists and reports.

6.5 THE EFFECTS OF PRODUCT MODEL BASED APPROACH ON THE PRODUCTIVITY AND QUALITY OF DESIGN WORK

The feedback gained from Xbuild users show that the software helps to improve both the productivity and the quality of design work. The increased productivity means basically that a structure can be designed and detailed faster and using less manhours than by conventional means. The increase of productivity varies quite much from project to project. In some cases, when the user is designing "standard solutions" that vary just a little from each other, the saving in manhours has been up to 80%.

The increased quality of design and detailing can be noticed on building sites. In most projects where Xbuild software has been used, the amount of design errors has been proved to be zero. This is mainly due to the fact that when all design information is stored in one place - the product model - the possibility of incoherent information is practically non-existing.

```
#include "e3user.h"

int joint_28(joint_t *joint, joint_parameter_t par,
                screw_parameter_t screw,welding_parameter_t welding)
{
/*****************************************************************/
/* NOSTURIKONSOLI_1                                              */
/* This part of the joint is the actual creation of the joint    */
/*****************************************************************/

/* declaration of internal variables */
    int                  result = 1, id_console = 0;
    joint_parameter_t    sub_par;
    joint_t              sub_joint;
    point_               *p3;
    xs_line_t            line;
    xs_coord_system_t    pcoord, scoord;
    profile_data_t       sdata, pdata, kdata, prdata;
    xs_welding_position_t welding_position;
    welding_parameter_t sub_welding;
/* Parameters */
/*****************************************************************/
/* fixed parameters */
    par.bpl[1] = DEF;

/* default values */
    if(screw.name[0] == 0) strcpy(screw.name, "M");
    if(screw.diameter == DEF) screw.diameter = 20;
    if(screw.nw == DEF) screw.nw = 2;
    if(screw.lwtyp == DEF) screw.lwtyp = 2;
    if(screw.nb == DEF) screw.nb = 2;
    if(screw.lbtyp == DEF) screw.lbtyp = 2;
    if(par.r1 == DEF) par.r1 = 10.0;
    if(par.r2 == DEF) par.r2 = 10.0;
    if(par.tpl[1] == DEF) par.tpl[1] = 20.0;
    if(par.hpl[1] == DEF) par.hpl[1] = 50.0;
    if(par.cprof[0] == 0) strcpy(par.cprof, "HE300A");
    if(par.tol[1] == DEF) par.tol[1] = 0.0;
    if(par.tol[2] == DEF) par.tol[2] = 0.0;
    if(par.tol[3] == DEF) par.tol[3] = 0.0;

/* Now we are looking at the joint with the beam running horizontally
   from left to right */

/* Get the size and position of the beams */

    xs_profile_data(joint->prim, &pdata);
    xs_profile_data(joint->sec[0], &sdata);
    if( !xs_section_data(par.cprof, &kdata) ) XS_ERROR(FATAL,10);

/* Default values */

    par.tpl[1] = xs_plate_thickness( screw.diameter/2, par.tpl[1]);
    par.tj[1]  = xs_plate_thickness( 1.5*pdata.s, par.tj[1]);
    par.tj[2]  = xs_plate_thickness( 1.5*kdata.s, par.tj[2]);
    par.tj[3]  = xs_plate_thickness( 1.5*sdata.s, par.tj[3]);

/* Do some checkings */

    if((screw.nw * screw.nb) > MAXSCREW) XS_ERROR(FATAL,0);
```

```
    /* Beam position checks */

    xs_part_coord_system(joint->prim, &pcoord);
    xs_part_coord_system(joint->sec[0], &scoord);

    if(fabs(pcoord.p[2]->x - pcoord.p[0]->x) > 10.0) XS_ERROR(FATAL,24);
    if(fabs(scoord.p[2]->z - scoord.p[0]->z) > 10.0) XS_ERROR(FATAL,25);
    if(fabs(pcoord.p[0]->z) - pdata.h/2-sdata.b/2- (par.r1)- (par.r2) < 10.0)
        XS_ERROR(FATAL,26);
    if(MAX(kdata.b, kdata.b2) > MIN(pdata.b, pdata.b2)) XS_ERROR(WARNING,27);

/* Changing to right plane depending on which side the beam is */

    if(pcoord.p[0]->z > 0.001)
    {
        xs_point(&line.p1, 0.0, 0.0, 0.0);
        xs_point(&line.p2, 0.0, 0.0, -1000.0);
        xs_point(&p3, 0.0, 1000.0, 0.0);
    }
    else
    {
        xs_point(&line.p1, 0.0, 0.0, 0.0);
        xs_point(&line.p2, 0.0, 0.0, 1000.0);
        xs_point(&p3, 0.0, 1000.0, 0.0);
    }
    xs_set_plane(line.p1, line.p2, p3);

/* Now we are looking at the joint with the beam perpendicular to the plane
   and on the left side of the column */

/* Generate console beam */

    xs_point(&line.p1, pcoord.p[0]->x+pdata.h/2, -sdata.h/2-par.tpl[1], 0.0);
    xs_point(&line.p2, sdata.b/2, -sdata.h/2-par.tpl[1], 0.0);

    xs_part_attributes(joint->pos1, "KONSOLI", joint->mat,
                       joint->group, joint->group2);
    xs_position_attributes(0.0, par.r1 + par.r2,
            RIGHT, 0.0, FRONT, 0.0, MIDDLE, 0.0);
    id_console = xs_beam(par.cprof, line);

    if (id_console && (welding.type[2] || welding.type2[2])) /*prim top */
    {
        xs_create_welding_by_settings(&welding, 2, kdata.t, pdata.t, kdata.t,
                                      WELD_DIR_PLUS_Y, joint->prim, i
    }

    if (id_console && (welding.type[2] || welding.type2[2])) /*prim bottom */
    {
        xs_create_welding_by_settings(&welding, 2, kdata.t2, pdata.t, kdata.t2
                                      WELD_DIR_MINUS_Y, joint->prim,
    }

    if (id_console && (welding.type[3] || welding.type2[3])) /*prim web */
    {
        xs_create_welding_by_settings(&welding, 3, kdata.s, pdata.t, kdata.s,
                                      WELD_DIR_PLUS_Z, joint->prim, i
    }

/* upper column stiffeners */

    xs_profile_data(id_console, &prdata);

    sub_par = par;
    sub_par.tpl[1] = par.tj[1]; sub_par.bpl[1] = par.bj[1]; sub_par.hpl[1] = p
```

```
    sub_par.tpl[2] = par.tj[1]; sub_par.bpl[2] = par.bj[1]; sub_par.hpl[2] = pa
    sub_par.tol[1] = par.tol[1];
    sub_welding = welding;
    xs_copy_welding_attributes(&welding,0, &sub_welding,1);
    xs_copy_welding_attributes(&welding,1, &sub_welding,2);

    sub_joint = *joint;
    xs_point(&sub_joint.reference_point, pcoord.p[0]->x,
             -sdata.h/2 - par.tpl[1] - prdata.t/2, 0.0);
    xs_point(&sub_joint.xdir, 0.0,  -1000.0, 0.0);
    xs_point(&sub_joint.up, 1000.0, 0.0, 0.0);

    if( !xs_detail(&sub_joint, STIFFENER, sub_par, screw, sub_welding )) resul

/* lower column stiffeners */

    sub_par.tpl[1] = par.tj[1]; sub_par.bpl[1] = par.bj[1]; sub_par.hpl[1] = pa
    sub_par.tpl[2] = par.tj[1]; sub_par.bpl[2] = par.bj[1]; sub_par.hpl[2] = pa
    sub_par.cut1 = 1; sub_par.cut2 = 0; sub_par.tol[1] = par.tol[1];

    xs_point(&sub_joint.reference_point, pcoord.p[0]->x,
             -sdata.h/2 - par.tpl[1] - prdata.h + prdata.t2/2, 0.0);
    xs_point(&sub_joint.xdir, 0.0,  - 1000.0, 0.0);
    xs_point(&sub_joint.up, 1000.0 , 0.0, 0.0);

    if( !xs_detail(&sub_joint, STIFFENER, sub_par, screw, sub_welding )) resul

/* make crossing between console and beam */

    sub_par = par;
    sub_par.dist = DEF;        /* cutdistance of secondary beam, now do not cut
    sub_par.tpl[1] =  DEF;       /* thickness of t-piece plates*/
    sub_par.bpl[1] = sdata.b2 + 2* par.r1; /* width of t-piece plates */
    sub_par.hpl[1] = par.hpl[1];/* height of t-piece plates(sec. beam directio
    sub_par.tpl[2] =  par.tj[2];       /* thickness of stiffeners in mainbeam*/
    sub_par.bpl[2] =  par.bj[2];       /* width of stiffeners in mainbeam*/
    sub_par.hpl[2] =  par.hj[2];       /* height of stiffeners in mainbeam*/
    sub_par.tpl[3] =  par.tj[3];       /* thickness of stiffeners in sek.beam*/
    sub_par.bpl[3] =  par.bj[3];       /* width of stiffeners in sek.beam*/
    sub_par.hpl[3] =  par.hj[3];       /* height of stiffeners in sek.beam*/
    sub_par.tpl[4] =  0;             /* thickness of stiffeners in t-piece */
    sub_par.tol[1] = par.tol[2];       /* tolerance of stiffeners in mainbeam
    sub_par.tol[2] = par.tol[3];       /* tolerance of stiffeners in sek.beam *
    sub_par.cut = 0;

    sub_joint = *joint;
    sub_joint.prim = id_console;
    sub_joint.xdir = DEFAULT_DIRECTION;
    xs_point(&sub_joint.up, 0.0, 1000.0, 0.0);
    sub_joint.position_type = MIDDLE_PLANE;
    sub_welding = welding;
    xs_copy_welding_attributes(&welding,4, &sub_welding,0);
    xs_copy_welding_attributes(&welding,4, &sub_welding,1);
    xs_copy_welding_attributes(&welding,5, &sub_welding,2);
    xs_copy_welding_attributes(&welding,6, &sub_welding,3);
    xs_copy_welding_attributes(&welding,6, &sub_welding,4);
    xs_copy_welding_attributes(&welding,7, &sub_welding,5);
    xs_copy_welding_attributes(&welding,8, &sub_welding,6);
    xs_copy_welding_attributes(&welding,9, &sub_welding,7);

    if( !xs_joint(&sub_joint, CROSSING, sub_par, screw, sub_welding) ) result

    return result;
}
```