Integrating LSTC and MSC.Software Technology for Explicit Dynamics and Fluid-Structure Interaction

Author:

Erik Plugge, MSC.Software Benelux B.V.

Correspondence:

MSC.Software Benelux B.V.
Groningenweg 6
2803 PV – Gouda
The Netherlands

Telephone: +31-(0)182-536 444
Fax: +31-(0)182-538 418
Erik.Plugge@mscsoftware.com

Keywords:
MSC.Nastran, SOL 700, LS-DYNA, MSC.Dytran, Explicit Dynamics, FSI

ABSTRACT

LSTC and MSC.Software have entered a long-term strategic partnership, with the objective of integrating complementary technology to the benefit of the structural and fluid dynamics community. This paper presents some of the technical aspects of this integration, highlights the technologies being used, and demonstrates the new application capabilities being enabled by this partnership.
The presentation will focus on two aspects of code coupling. Firstly, between MSC.Software's MSC.Dytran Eulerian solver and LSTC's LS-DYNA explicit solver, and secondly the integration of LS-DYNA explicit solutions via the SOL700 sequence in MSC.Nastran.
Both the Eulerian solver in MSC.Dytran and the structural explicit solver in LS-DYNA are unique in their class. Coupling the two codes together allows FSI (Fluid-Structure Interaction) simulations to be run efficiently and on a high level of accuracy and speed. Similarly, coupling of the LS-DYNA explicit solver with MSC.Nastran via SOL700 enables a new range of applications in crash and impact from within a general-purpose structural environment.
This paper describes technical aspects of how these code integrations have been developed, and presents a range of application areas and examples of the new dynamics and fluid-structure interaction capabilities achieved.

INTRODUCTION

The individual source code components needed for the integration are defined as follows:

- MSC.Dytran (MSC.Software): Explicit Finite Volume/Finite Element solver, strong in the coupling between CFD and Structural solver: called Fluid-Structure Interaction (FSI).
- LS-DYNA (LSTC): Explicit Finite Element solver, strong in the speed-optimized Structural solver: LS-DYNA DMP, specifically crash-like applications.
- MSC.Nastran (MSC.Software): Implicit Structural solver, well established in the automotive and aerospace industry for linear static and dynamics.

MSC.Nastran Explicit Nonlinear (SOL 700) is an application module based on the Dytran LS-DYNA solver that offers a powerful explicit solution to analyze a variety of problems involving short duration, highly dynamic events with severe geometric and material non-linearities. Reference [1] describes the available possibilities in MSC.Nastran 2004r3, which was the first prerelease of this solution sequence.

The solution sequence SOL 700 is being developed in phases. The following flow diagram summarizes the dataflow in a MSC.Nastran SOL700 simulation for phase 1. Here the LS-DYNA solver is invoked for pure structural analyses.

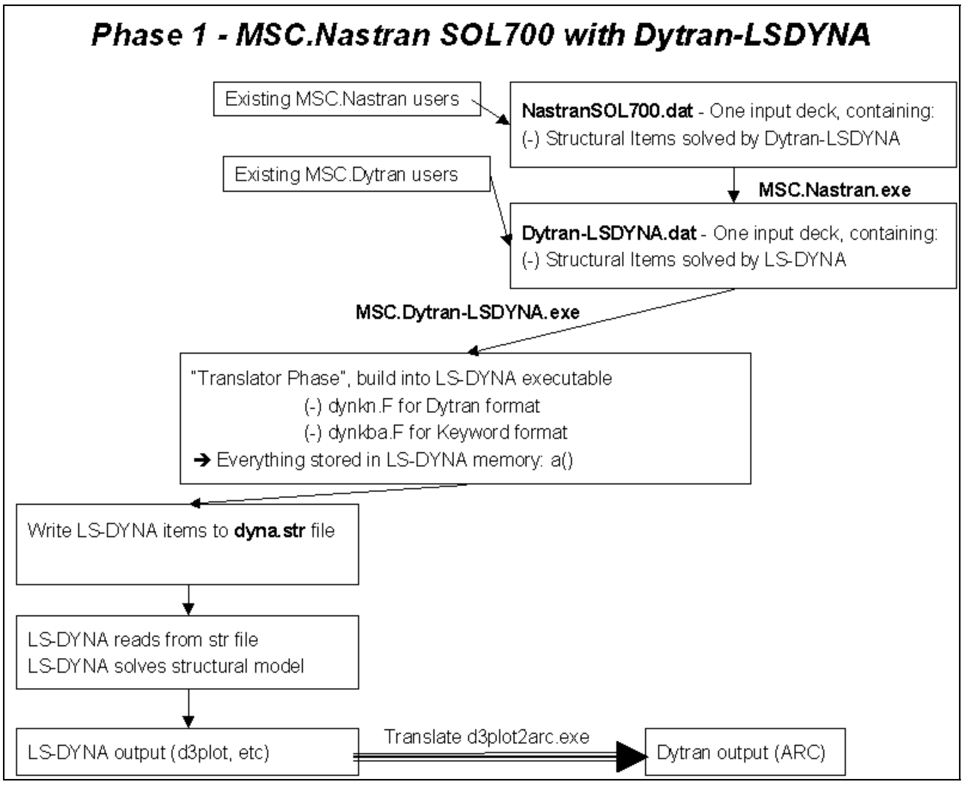

Figure 1. MSC.Nastran SOL700 - phase **1**

The second phase and beyond includes advanced fluid-structure interaction capabilities. During this phase both the structural LS-DYNA solver and the MSC.Dytran FSI solver will be invoked. This is schematically shown in figure 2.

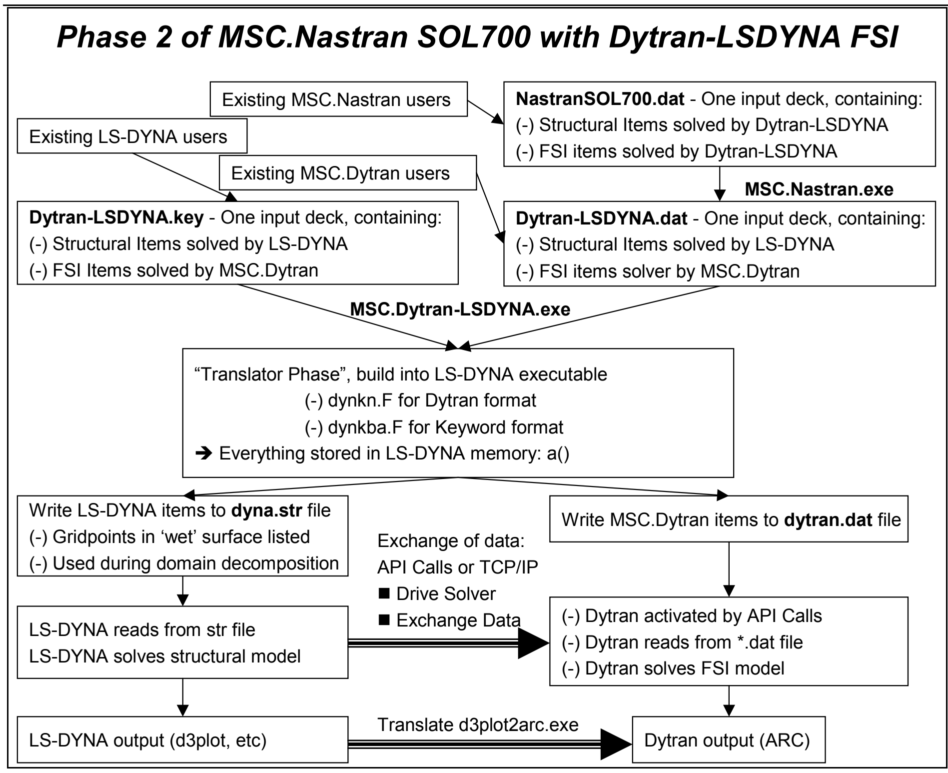

Figure 2. MSC Nastran SOL700 – phase 2.

As is shown in figure 2 three types of users will benefit from this partnership between MSC.Software and LSTC:

- Existing LS-DYNA users, since they can access MSC.Dytran's FSI through the familiar keyword format.
- Existing MSC.Nastran users, since they can access an Explicit Non-Linear solution from the familiar MSC.Nastran bulk data format.
- Existing MSC.Dytran users since they can access the LS-DYNA structural solver through the familiar MSC.Dytran format.

The following chapters will describe the different phases behind the flow diagram above:

- Preprocessing
- Solvers and Technology
- Post processing

Finally a few examples of the potential usage of the technology described will be given.

MSC.Nastran SOL700 - Preprocessing

In all three user environments an input deck is the start for a simulation.

Either a LS-DYNA keyword file, a MSC.Nastran input file or a MSC.Dytran input file can be used.

Any graphical user interface that supports these formats can be used for preprocessing. The phase 2 FSI project focuses on the input formats of the MSC.Nastran input file and LS-DYNA keyword file. Currently MSC.Patran 2005r2 supports MSC.Nastran SOL700 (phase 1) completely.

Solvers and Technology

SOL700 – Phase 1

The MSC.Dytran LSDYNA project in phase 1 merely focused on mapping the MSC.Dytran input file format into LS-DYNA memory, the so-called a() and ia() arrays. This mechanism is meanwhile smoothly and completely available in the production version.

SOL700 – Phase 2

In phase 2 the MSC.Dytran code and the LS-DYNA code have been coupled to a very high extent. Development is extended to support the FSI related definitions in the MSC.Nastran input file format. The FSI related definitions in MSC.Nastran input format are simply mapped to the MSC.Dytran input file format, which are subsequently picked up by the MSC.Dytran LS-DYNA solver and mapped on to the internal arrays. The latter also makes it possible to fill the internal arrays with data from the keyword file and thus obtain full support of future FSI simulations underneath LS-DYNA.

Once all the input data is stored in the internal data arrays in LS-DYNA a decision is made whether the data is needed for the structural or fluid part in the simulation. The structural items will remain inside LS-DYNA, the fluid part will be written to an intermediate input file for MSC.Dytran. The items that are processed and used in both solvers are part of the so-called "wet surface". These items remain both in LS-DYNA and are written to the intermediate input file for MSC.Dytran.

The model used to do the data exchange is a client-server model, where LS-DYNA is the client which steers the MSC.Dytran server. This has the advantage that the coding in both solvers remains transparent because there will be no flag-driven logic or callback functions necessary. In order to make a client-server model like this possible a flat code structure is required in the MSC.Dytran source code. A big advantage of this flat code structure is that the so-called Application Program Interface (API) drivers, can be published to the outside world in order for a user to create a custom-build executable, see figure 3.

```
Flat Dytran structure

Program dytran
          call mscd_api_p0_set_api_type()
          call mscd_api_p0_init_memory()
          call mscd_api_p1_driver()
          call mscd_api_p2_driver()
          call mscd_api_p3_driver()
C    Start cycle
 100      continue
          call mscd_api_p4_driver_move_points()
          call mscd_api_p4_driver_contact()
          call mscd_api_p4_driver_couple()
          call mscd_api_p4_driver_lagpro()
          call mscd_api_p4_driver_eulpro()
          call mscd_api_p4_driver_get_time()
          call mscd_api_p4_driver_set_time()
          call mscd_api_p6_driver()
C Goto next cycle
          goto 100
```

Figure 3. Flat code structure in MSC.Dytran

Next to the usage of the API in MSC.Dytran LS-DYNA, it has also been used successfully in the coupling between MSC.Adams and MSC.Dytran in a fuel tank sloshing analysis. The MSC.Dytran LSDYNA integration for FSI is using the flat code structure as follows (figure 4)

```
Flat MSC.Dytran structure called from LS-DYNA

Program MSC.Dytran-LSDYNA
     | …. Initialization of LSDYNA
     call mscd_api_p0_set_api_type()
     call mscd_api_p0_init_memory()
     call mscd_api_p1_driver()
     call mscd_api_p2_driver()
     call mscd_api_p3_driver()
     | …. Initialization of LSDYNA
C
C    Main driver routine in LSDYNA:
     subroutine fem3d
     100     continue
             | … update nodal positions…
             call mscd_api_set_gp_data()          (fem3d sets gp positions in Dytran)
             | … fem3d does usual stuff, like contact & elements
             call mscd_api_p4_driver_couple()     (fem3d tells Dytran to update the FSI-coupling)
             call mscd_api_p4_driver_eulpro()     (fem3d tells Dytran to update the FSI-solver)
             |… fem3d does usual stuff
             call msc_api_get_gp_data()           (fem3d gets nodal forces of FSI)
             |… add nodal forces to LS-DYNA arrays
             call mscd_api_p4_driver_get_time()
             | … determine stable timestep
             call mscd_api_p4_driver_set_time()   (fem3d sets time-step in Dytran)
             |… -next cycle
             goto 100
```

Figure 4. Flat MSC.Dytran structure called from LS-DYNA

In figure 4 LS-DYNA is the main driver of the solution. Thus all API-calls to MSC.Dytran are done from LS-DYNA. The first set of mscd_api calls will initialize MSC.Dytran just like a normal simulation: it uses the intermediate MSC.Dytran input file written when dividing the entries in the original input file. As mentioned before this original input file can be a LS-DYNA keyword file or a MSC.Dytran input file.

MPP considerations
In phase 1 of SOL700 the functionality of the MSC.Dytran solver is not included yet. This phase gives the user access to the fast structural MPP possibilities available in LS-DYNA.
In phase 2 of SOL700 part of the FSI capabilities of MSC.Dytran are supported. The LS-DYNA solver runs in MPP mode if requested by the user. Initially the MSC.Dytran part will only run in single processor mode on the master CPU or another dedicated CPU. The master CPU will call the API functions of MSC.Dytran. This method requires a gather and scatter logic on the LS-DYNA side: the master CPU will gather all the gridpoint positions in LS-DYNA and sets them into MSC.Dytran by means of the API call to mscd_api_set_gp_data. Subsequently the master CPU in LS-DYNA will scatter the additional forces on the wet-surface to the slave CPU's once they have been retrieved from MSC.Dytran.
In later phases of SOL700 the MSC.Dytran part will also run in MPP mode. The MSC.Dytran part will use it's own decomposition, which can be driven by LS-DYNA's structural decomposition. The latter is important to obtain minimal communication.

API considerations
In the prototype of MSC.Dytran LS-DYNA-FSI the two codes are communicating by means of direct Fortran calls from LS-DYNA. This requires the LS-DYNA and MSC.Dytran libraries to be linked into one executable. The main advantage of this is that the performance is least affected. The big disadvantage is the requirement of matching compile and link flags on each supported platform. This means that upgrades to new compiler versions and/or hardware Operating Systems are a big burden on maintenance.
An alternative for the current communications is to build an extra layer into the API. This extra layer will couple the two codes together by means of TCP/IP communication. This type of communication has been used in coupling the MSC.Dytran and MSC.Adams codes.
A third communication possibility is to include the FSI source code into LS-DYNA's main memory. The advantage is that MPP considerations become easier: the MSC.Dytran part can make use of the domain decomposition as is used by the LS-DYNA Euler solver. This third communication possibility still requires more investigation.

Fluid Structure Interaction in MSC.Dytran
Over the past years some breakthrough technology has been added to the FSI solver of MSC.Dytran. With reference to [2] through [7] a short summary is given here on the technology that becomes accessible to MSC.Dytran LS-DYNA and MSC.Nastran SOL700 users.
Fluid dynamics in a Eulerian frame of reference is described by the conservation laws of mass, momentum and energy in a volume V that is bounded by a surface A with surface normal n

$$\frac{d}{dt}\int_V \rho dV + \int_A \rho(u \cdot n)dA = 0$$

$$\frac{d}{dt}\int_V \rho u_i dV + \int_A \rho u_i (u \cdot n)dA = -\int_A p n_i dA \qquad \text{(Equations 1-3)}$$

$$\frac{d}{dt}\int_V \rho e dV + \int_A \rho e(u \cdot n)dA = -\int_A u_i p n_i dA$$

Here p and ρ are respectively pressure and density; u is the velocity vector and e the total energy per unit mass. The pressure in Eulerian fluid material is either given by the ideal gas law, or by a polynomial function of density.

The changes in mass, momentum and energy of an Euler element occurring in a time step follow by applying the conservation laws to the effective volume of the Euler element. Surface integrals are computed by integration across the effective boundary of the Euler element and represent the amount of mass, momentum and energy that is transported through the surface. The procedure used to compute the transported amounts and the time integration determine the order of the accuracy. MSC.Dytran contains a number of different solvers that are described in detail in reference [7].

In order to make a FSI simulation possible the material inside an Euler mesh must be able to interact with an element or body, called shell-surface here after. The shell acts as a barrier to the material in the Eulerian domain and therefore determines the pressures inside the Euler elements. On the other hand the Eulerian material exerts a force on the shell surface that can result in a displacement and deformation of the shell. This force is deduced from the pressure in the Euler elements. The shell surface is either modeled by shell/membrane elements or assumed to be rigid. If a shell surface acts as barrier to an Euler domain the surface and it's elements will be called respectively a coupling surface and segments. Material can be either inside or outside the coupling surface. In the following we will assume that the material is inside the coupling surface.

Element pressure readily follows from the density. For density computations the volume of each Euler element is required. Only that part of the element that falls within the coupling surface can contain mass and can contribute to this volume. The boundary of this effective volume is shown in figure 5. It consists of interfaces between Euler elements and parts of segments. Only those parts of segments that are within the Euler element contribute to this boundary. Therefore a segment needs to be partitioned in sub segments such that every sub segment is in exactly one segment and in exactly one Euler element. This partitioning enables a straightforward computation of volume and is also used to transfer the force that the material exerts on the structure to the shell elements.

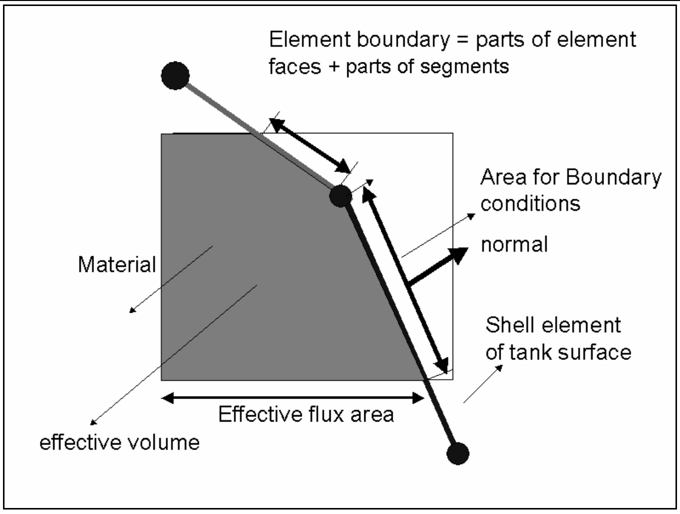

Figure 5. The effective boundary of an euler element

In the coupling of MSC.Dytran's FSI solver with the structural LS-DYNA solver it is clear that the Eulerian force on the shell-surface has to be communicated to LS-DYNA. LS-DYNA will add the Eulerian force to the internal and external forces like contact and boundary forces. Finally the total force will be used to update the shell surface in LS-DYNA, after which the new position of the shell surface is communicated back to MSC.Dytran.

Fluid-structure interaction with failure in MSC.Dytran

Consider a box filled with gas. If a blast wave is initiated inside the box some parts of the box may fail and gas can escape through ruptures. To simulate this flow, the gas inside the box is modeled by an Euler domain and the box surface by shell elements. These shell elements form the coupling surface for this Euler domain. Once shell elements of this box have failed, gas flows from the inner domain to an outer Euler domain that models the ambient. The shell surface will also form the coupling surface for this outer Euler domain and is therefore able to connect Euler elements inside the shell surface to elements outside the shell surface. Flow from one Euler domain to another is also possible through fully or partly porous segments.

FSI with failure is part of the next phases of the SOL700 development work.

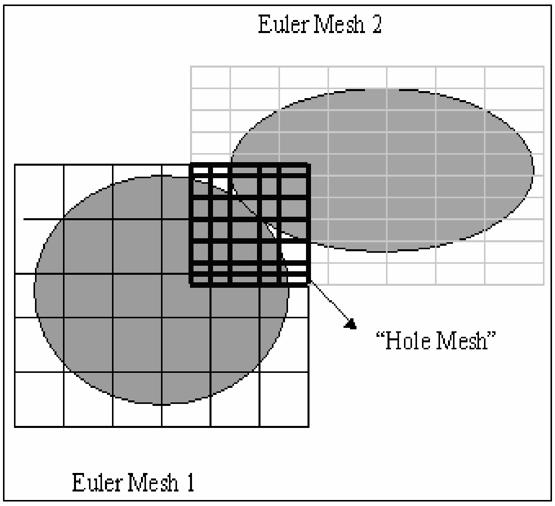

Figure 6 The overlapping mesh.

Flow between Domains in MSC.Dytran

Flow from one Euler domain to another takes place through either shell elements that are porous or that have failed. Flow through a segment can only take place if it is inside both Euler domains. In the following, let us assume that both Euler domains are sufficiently large. In general a segment can intersect several Euler elements of the first Euler domain and the same applies to the second domain. Therefore the segment connects several Euler elements in the first Euler domain to several Euler elements in the second domain. For an accurate and straightforward computation of flow through the segment, it is partitioned into sub segments such that each sub segment is in exactly one Euler element of each Euler domain. To carry out this partitioning an overlapping Euler mesh is created, that is the union of both Euler meshes. Then, for each element in this overlapping domain the intersection with the segment is determined. Each intersection gives one sub segment that refers to both the original segment and to the element in the overlap domain. Since an element in the overlap domain is in exactly one element of both domains the sub segment connects exactly one element in the first Euler domain to exactly one element in the second domain. Transport across this sub segment is straightforward because it closely resembles transport that takes place between the Euler elements that are within an Euler domain. In computing transport across the sub segment the velocity of the segment has to be taken into account. If the segment is moving with the same velocity as the material on either side no material will flow through the segment.
FSI with failure is part of the next phases of the SOL700 development work.

MSC.Nastran SOL700 – Post processing

In all user environments it is possible to, either use the LS-DYNA native d3plot and d3thdt output files, or use the output files common to that environment. MSC.Nastran will support the typical MSC.Nastran output formats, like DBALL, OP2 and PCH. All of these formats are widely supported by commercially available products, including LS-POST, CEI/Insight and MSC.Patran. The images in Figure 11 and 12 are created with CEI/Ensight

MSC.Nastran SOL700 – Examples

SOL 700	TARGET INDUSTRIES			
	Auto	Aero	Defense	Consumer
Applications	Crash	Crashworthiness	Impact	Drop Test
	Bumpers	Engines Design	Penetration - Perforation	Bottles
	Component Crush – Energy Absorbtion	Containment Analysis	Survivability*	BioMed
	Tire Performance*	Tri-Hub Burst – Blade Out	Blast – Explosives*	Containers
	Fuel Tanks*	Bird Strike*	HRAM*	Composites
	Fuel Pumps*	Seats Design*	UNDEX*	Papers*
	Airbags & OS*	Fuel Tanks – HRAM*	Weapon Design*	Boats*

* These are FSI Applications and will be part of Phase II & III

Figure 7. Target applications for MSC SOL700

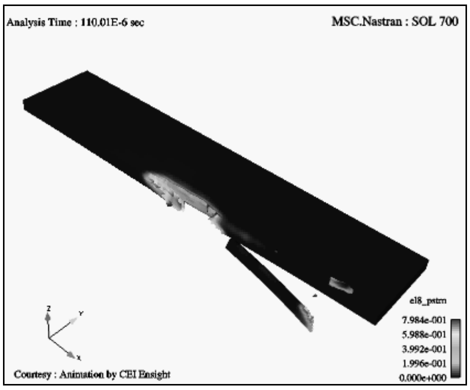

Figure 8. Impact (*MSC Nastran SOL700 – phase 1*)

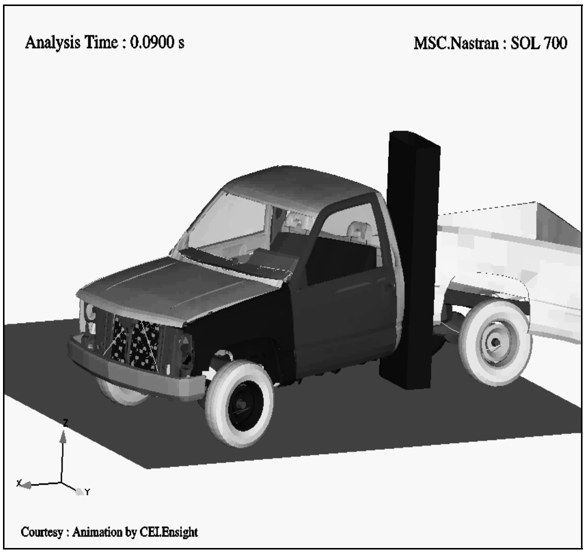

Figure 9. Side crash (MSC.Nastran SOL700 – phase 1)

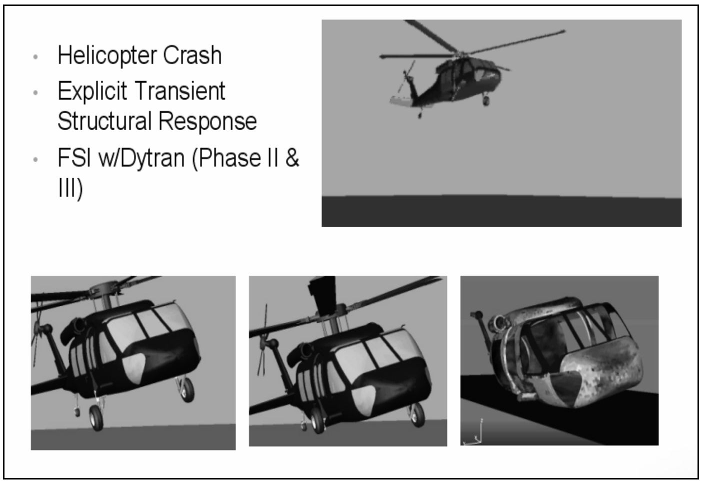

Figure 10. Helicopter crash on ground (MSC.Nastran SOL700 – phase 1)
Helicopter crash on water (MSC.Nastran SOL700 – phase 2)

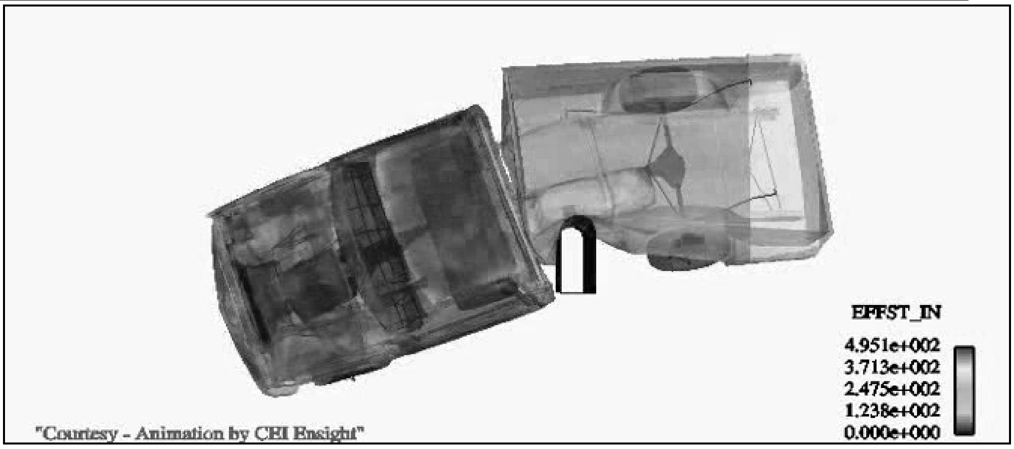

Figure 11. Fuel tank in truck side crash (MSC.Nastran SOL700 – phase 2)

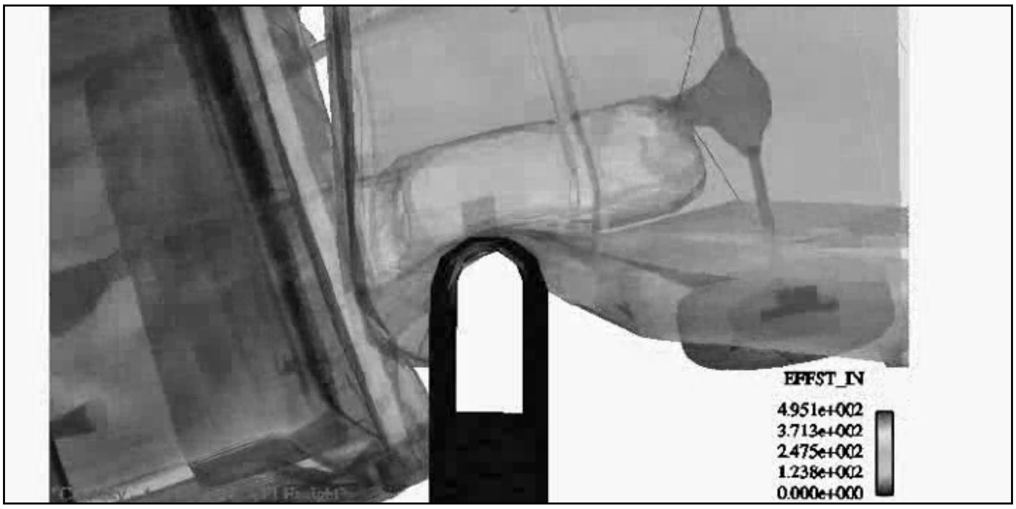

Figure 12, Fuel tank in truck side crash (MSC.Nastran SOL700 – phase 2)

Summary and Conclusions

This paper describes the technical aspects of the coupling of the two explicit analysis codes LS-DYNA and MSC.Dytran and the different environments in which this solution is made available. The familiar environments for either LS-DYNA or MSC.Nastran users, is key to the success of the integration.
The MSC.Dytran code now contains an API that opens up the Eulerian and Coupling processors. The MPP implementation in the structural LS-DYNA code together with the FSI capabilities of MSC.Dytran will make a range of application areas accessible to users who are using the de-facto industry standards, LS-DYNA and MSC.Nastran.

References

1. Doelfs P, Neubauer I; Using MSC.Nastran for Explicit FEM Simulations. 3. LS-DYNA Anwenderforum, Bamberg 2004, 2004
2. Edwin de Vries, A general coupling algorithm for the simulation of fluid-structure interaction. APUC MSC.Software Asia Pacific User Conference, 2000
3. Van der Veen W.A, Simulation of a compartmented airbag deployment using an explicit, coupled Euler/Lagrange method with adaptive Euler domains. NAFEMS, Florida, March 2003

4. Van der Veen W.A, Simulation of fluid flow initiated by failure of structural parts using an explicit coupled Euler/Lagrange method. To appear in the proceedings of the NAFEMS 2005 World congress, 2005
5. Kamiya, Yamaguchi, de Vries, Simulation Studies of Sloshing in a fuel tank, SAE 2002, 2002
6. Ding P, Buijk A.J, Van der Veen W.A, Simulation of Fuel Tank Filling using a Multi-material Euler Solver with Multiple, Adaptive Domains. To appear in the proceedings of the SAE 2005 World congress, 2005
7. Van der Veen W.A, Crushing simulation of a partially filled fuel tank beyond failure. To appear in the proceedings of the SAE 2005 World congress, 2005