Earth Explorer CFI: Mapping of V2.X.X functions to V3.X.X functions in user applications

1. GENERAL OBSERVATIONS

Most of the interface changes between the versions 2.X.X and 3.X.X of the Earth explorer CFI software are due to the use of a new variable type called CFI identifier (from now on, the CFI identifiers will be noted as *Ids*). Different kinds of Ids have been created to reflect the different categories or "objects" that group the data handled in the CFI. This means that each Id type stores internal data needed for a specific computation. As a consequence, the dependencies between the different Ids have been identified and the CFI routines have been modified accordingly. Note that the Ids follow a hierarchical structure. A list of the Ids used in the CFI is given in the table below:

Id	Library	Description	Usage	Dependencies
sat_id	-	Satellite identifier	Input parameter (does not need to be initialised)	Independent, no previous initialisation of any other Id is required
xl_time_id	explorer_lib	It stores the time correlations	Output parameter (when initialised) and Input parameter in time related computations and time closing function	Independent, no previous initialisation of any other Id is required
xo_orbit_id	explorer_orbit	It stores the orbit data needed for orbit calculations	Output parameter (when initialised) and Input parameter in orbit and visibility related computations and orbit closing function	xo_orbit_id = sat_id + time_id + (orbit data). It requires that xl_time_id has been previously initialised
xo_propag_id	explorer_orbit	It stores the data needed for orbit propagation	Output parameter (when initialised) and Input parameter in propagation routines and propagation closing function	xo_propag_id = orbit_id + (propagation data). It requires that xl_orbit_id has been previously initialised
xo_interpol_id	explorer_orbit	It stores the data needed for interpolation calculations	Output parameter (when initialised) and Input parameter in interpolation routines and interpolation closing function	xo_interpol_id = orbit_id + (interpolation data). It requires that xl_orbit_id has been previously initialised

Id	Library	Description	Usage	Dependencies
xp_atmos_id	explorer_pointing	It stores the atmospheric data used in target functions	Output parameter (when initialised) and Input parameter in target routines and atmospheric closing function	Independent, no previous initialisation of any other Id is required
xp_dem_id	explorer_pointing	It stores the Digital Elevation Model data used in target functions	Output parameter (when initialised) and Input parameter in target routines and DEM closing function	Independent, no previous initialisation of any other Id is required
xp_sat_nom_trans_id	explorer_pointing	It stores the Satellite Nominal Attitude Ref Frame data used in attitude functions	Output parameter (when initialised) and Input parameter in attitude routines and satellite nominal attitude transformation closing function	Independent, no previous initialisation of any other Id is required, except when the file initialisation routine is used. Then xp_sat_nom_trans_id requires that xl_time_id has been previously initialised
xp_sat_att_trans_id	explorer_pointing	It stores the Satellite Attitude Ref Frame data used in attitude functions	Output parameter (when initialised) and Input parameter in attitude routines and satellite attitude transformation closing function	Independent, no previous initialisation of any other Id is required, except when the file initialisation routine is used. Then xp_sat_att_trans_id requires that xl_time_id has been previously initialised
xp_instr_trans_id	explorer_pointing	It stores the Instrument Ref Frame data used in attitude functions	Output parameter (when initialised) and Input parameter in attitude routines and instrument transformation closing function	Independent, no previous initialisation of any other Id is required, except when the file initialisation routine is used. Then xp_instr_trans_id requires that xl_time_id has been previously initialised
xp_attitude_id	explorer_pointing	It stores the results of the attitude calculation used in target functions	Output parameter (when initialised) and Input parameter in target routines	xp_attitude_id = xl_time_id + xp_sat_nom_trans_id + xp_sat_att_trans_id +

Id	Library	Description	Usage	Dependencies
			and attitude closing function	xp_instr_trans_id + attitude computation. It requires that xl_time_id has been previously initialised but it does not necessary require that xp_sat_nom_trans_id, xp_sat_att_trans_id and xp_instr_trans_id have been previously initialised
xp_target_id	explorer_pointing	It stores the results of the target calculation, needed to get ancillary results	Output parameter (when initialised) and Input parameter in extra results target routines and target closing function	xp_target_id = xp_attitude_id + xp_atmos_id + xp_dem_id + target data. It requires that xp_attitude_id has been previously initialised but, it does not necessary require that xp_dem_id and xp_atmos _id have been previously initialised

- The time correlations need to be initialised previously before performing any orbit calculation.
- There is a common set of orbit initialization routines regardless of what type of orbit calculations that are going to be performed: propagation, interpolation, time2orbit or get some orbit info parameters.
- The basic rules to follow when initializing, using and closing *Id*s are described in Section 7.3 of the General SUM. Each explorer_XXX SUM details the calling interface and input/output parameters.

2. LIBRARY EXPLORER_LIB: EQUIVALENCES BETWEEN V2.3.X AND V3.7.2

It can be observed that the explorer_lib routines in V3.7.2 have lost the sat_id as input parameter when the computation carried out by the routine is independent of the satellite.
Color code:

- In red: Function parameters present in V2.3.X interface that are not part of V3.7.2 interface
- In green: New function parameters present in the interface of V3.7.2
- In grey: List of parameters that remain in V3.7.2
- In orange: Function whose name has been modified in V3.7.2

Routine name (V2.3)	V2.3	V3.7.2	Use
xl_time_ref_init_file	xl_time_ref_init_file(&trif_sat_id, &trif_time_model, &trif_n_files, trif_time_file, &trif_time_init_mode, &trif_time_ref, &trif_time0, &trif_time1, &trif_orbit0, &trif_orbit1, &trif_val_time0, &trif_val_time1, trif_ierr)	xl_time_ref_init_file(&trif_time_model, &trif_n_files, trif_time_file, &trif_time_init_mode, &trif_time_ref, &trif_time0, &trif_time1, &trif_orbit0, &trif_orbit1, &trif_val_time0, &trif_val_time1, &time_id, trif_ierr)	Initialization (Called once)
xl_time_ref_init	xl_time_ref_init(&tri_sat_id, tri_time, &tri_orbit_num, &tri_anx_time, &tri_orbit_duration, tri_ierr)	xl_time_ref_init(tri_time, &tri_orbit_num, &tri_anx_time, &tri_orbit_duration, &time_id, tri_ierr)	Initialization (Called once)
xl_time_ref_close	xl_time_ref_close (&trc_sat_id)	xl_time_close (&time_id, ierr)	Closing (Called once)
xl_change_cart_cs	xl_change_cart_cs (&sat_id, &calc_mode, &cs_in_t, &cs_out_t, &time_ref, &time_t, r_t, rd_t, r2d_t, r_out_t, rd_out_t, r2d_out_t)	xl_change_cart_cs(&time_id, &calc_mode, &cs_in_t, &cs_out_t, &time_ref, &time_t, r_t, rd_t, r2d_t, r_out_t, rd_out_t, r2d_out_t)	Calculation
xl_cart_to_geod	xl_cart_to_geod(&sat_id, &calc_mode, rr_t, rrd_t, &lon_t, &lat_t, &h_t, &lond_t, &latd_t, &hd_t)	xl_cart_to_geod(&calc_mode, rr_t, rrd_t, &lon_t, &lat_t, &h_t, &lond_t, &latd_t, &hd_t)	Calculation
xl_geod_to_cart	xl_geod_to_cart(&sat_id, &calc_mode, &lon_t, &lat_t, &h_t, &lond_t, &latd_t, &hd_t, rr_t, rrd_t)	xl_geod_to_cart(&calc_mode, &lon_t, &lat_t, &h_t, &lond_t, &latd_t, &hd_t, rr_t, rrd_t)	Calculation
xl_cart_to_kepl	xl_cart_to_kepl(&sat_id, pos_in, vel_in, &kepl_mode, kepl_out, ierr_ck)	xl_cart_to_kepl(pos_in, vel_in, &kepl_mode, kepl_out, ierr_ck)	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
xl_kepl_to_cart	xl_kepl_to_cart(&sat_id, &kepl_mode, kepl_in, pos_out, vel_out, ierr_kc)	xl_kepl_to_cart(&kepl_mode, kepl_in, pos_out, vel_out, ierr_kc)	Calculation
xl_geod_distance	xl_geod_distance(&sat_id, &lon1_t, &lat1_t, &lon2_t, &lat2_t, &h_t, &d_t, &az1_t, &az2_t)	xl_geod_distance(&lon1_t, &lat1_t, &lon2_t, &lat2_t, &h_t, &d_t, &az1_t, &az2_t)	Calculation
xl_sun	xl_sun(&sat_id, &time_ref, &time_ut1, rsun, rdsun, &local_err)	xl_sun(&time_id, &time_ref, &time_ut1, rsun, rdsun, &local_err)	Calculation
xl_moon	xl_moon(&sat_id, &time_ref, &time_ut1, rmoon, rdmoon, &local_err)	xl_moon(&time_id, &time_ref, &time_ut1, rmoon, rdmoon, &local_err)	Calculation
xl_planet	xl_planet(&sat_id, &planet_id, &time_ref, &time_ut1, rplanet, rdplanet, &local_err)	xl_planet(&time_id, &planet_id, &time_ref, &time_ut1, rplanet, rdplanet, &local_err)	Calculation
xl_star_radec	xl_star_radec(&sat_id, &time_ref, &time_ut1, &ra0, &dec0, &mu_ra, &mu_dec, &rad_vel, &par, &ra, &dec, &local_err)	xl_star_radec(&time_id, &time_ref, &time_ut1, &ra0, &dec0, &mu_ra, &mu_dec, &rad_vel, &par, &ra, &dec, &local_err)	Calculation
xl_time_transport_to_ascii	xl_time_transport_to_ascii(&t2a_sat_id, &t2a_trans_id_in, &t2a_time_ref_in, t2a_transport_in, &t2a_ascii_id_out, &t2a_time_ref_out, t2a_ascii_out, t2a_ierr)	xl_time_transport_to_ascii(&time_id, &t2a_trans_id_in, &t2a_time_ref_in, t2a_transport_in, &t2a_ascii_id_out, &t2a_time_ref_out, t2a_ascii_out, t2a_ierr)	Calculation
xl_time_transport_to_transport	xl_time_transport_to_transport(&t2t_sat_id, &t2t_trans_id_in, &t2t_time_ref_in, t2t_transport_in, &t2t_trans_id_out, &t2t_time_ref_out, t2t_transport_out, t2t_ierr)	xl_time_transport_to_transport(&time_id, &t2t_trans_id_in, &t2t_time_ref_in, t2t_transport_in, &t2t_trans_id_out, &t2t_time_ref_out, t2t_transport_out, t2t_ierr)	Calculation
xl_time_transport_to_processing	xl_time_transport_to_processing(&t2p_sat_id, &t2p_trans_id_in, &t2p_time_ref_in, t2p_transport_in, &t2p_proc_id_out, &t2p_time_ref_out, &t2p_processing_out, t2p_ierr)	xl_time_transport_to_processing(&time_id, &t2p_trans_id_in, &t2p_time_ref_in, t2p_transport_in, &t2p_proc_id_out, &t2p_time_ref_out, &t2p_processing_out, t2p_ierr)	Calculation
xl_time_processing_to_ascii	xl_time_processing_to_ascii(&p2a_sat_id, &p2a_proc_id_in, &p2a_time_ref_in, &p2a_processing_in, &p2a_ascii_id_out, &p2a_time_ref_out, p2a_ascii_out, p2a_ierr)	xl_time_processing_to_ascii(&time_id, &p2a_proc_id_in, &p2a_time_ref_in, &p2a_processing_in, &p2a_ascii_id_out, &p2a_time_ref_out, p2a_ascii_out, p2a_ierr)	Calculation
xl_time_processing_to_transport	xl_time_processing_to_transport(&p2t_sat_id, &p2t_proc_id_in, &p2t_time_ref_in, &p2t_processing_in, &p2t_trans_id_out,	xl_time_processing_to_transport(&time_id, &p2t_proc_id_in, &p2t_time_ref_in, &p2t_processing_in, &p2t_trans_id_out,	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
	&p2t_time_ref_out, p2t_transport_out, p2t_ierr)	&p2t_time_ref_out, p2t_transport_out, p2t_ierr)	
xl_time_processing_to_processing	xl_time_processing_to_processing(&p2p_sat_id, &p2p_proc_id_in, &p2p_time_ref_in, &p2p_processing_in, &p2p_proc_id_out, &p2p_time_ref_out, &p2p_processing_out, p2p_ierr)	xl_time_processing_to_processing(&time_id, &p2p_proc_id_in, &p2p_time_ref_in, &p2p_processing_in, &p2p_proc_id_out, &p2p_time_ref_out, &p2p_processing_out, p2p_ierr)	Calculation
xl_time_ascii_to_ascii	xl_time_ascii_to_ascii(&a2a_sat_id, &a2a_ascii_id_in, &a2a_time_ref_in, a2a_ascii_in, &a2a_ascii_id_out, &a2a_time_ref_out, a2a_ascii_out, a2a_ierr)	xl_time_ascii_to_ascii(&time_id, &a2a_ascii_id_in, &a2a_time_ref_in, a2a_ascii_in, &a2a_ascii_id_out, &a2a_time_ref_out, a2a_ascii_out, a2a_ierr)	Calculation
xl_time_ascii_to_transport	xl_time_ascii_to_transport(&a2t_sat_id, &a2t_ascii_id_in, &a2t_time_ref_in, a2t_ascii_in, &a2t_trans_id_out, &a2t_time_ref_out, a2t_transport_out, a2t_ierr)	xl_time_ascii_to_transport(&time_id, &a2t_ascii_id_in, &a2t_time_ref_in, a2t_ascii_in, &a2t_trans_id_out, &a2t_time_ref_out, a2t_transport_out, a2t_ierr)	Calculation
xl_time_ascii_to_processing	xl_time_ascii_to_processing(&a2p_sat_id, &a2p_ascii_id_in, &a2p_time_ref_in, a2p_ascii_in, &a2p_proc_id_out, &a2p_time_ref_out, &a2p_processing_out, a2p_ierr)	xl_time_ascii_to_processing(&time_id, &a2p_ascii_id_in, &a2p_time_ref_in, a2p_ascii_in, &a2p_proc_id_out, &a2p_time_ref_out, &a2p_processing_out, a2p_ierr)	Calculation
xl_time_add	xl_time_add(&sat_id, &proc_id, &time_ref, &processing_in_1, &processing_in_2, &processing_out, tad_ierr)	xl_time_add(&proc_id, &time_ref, &processing_in_1, &processing_in_2, &processing_out, tad_ierr)	Calculation
xl_time_diff	xl_time_diff(&sat_id, &proc_id, &time_ref, &processing_in_1, &processing_in_2, &processing_out, tdi_ierr)	xl_time_diff(&proc_id, &time_ref, &processing_in_1, &processing_in_2, &processing_out, tdi_ierr)	Calculation
xl_time_obt_to_time	xl_time_obt_to_time(&sat_id, &proc_id, &time_ref, &time0, obt0, &period0, obt, &time_out, o2t_ierr)	xl_time_obt_to_time(&sat_id, &proc_id, &obt_param, &obt_value_in, &time_out, o2t_ierr)	Calculation
xl_time_time_to_obt	xl_time_time_to_obt(&sat_id, &proc_id, &time_ref, &time_in, &time0, obt0, &period0, obt, t2o_ierr)	xl_time_time_to_obt(&sat_id, &proc_id, &obt_param, &time_in, &obt_value_out, t2o_ierr)	Calculation
xl_time_get_leap_second_info	xl_time_get_leap_second_info(&sat_id, &ascii_id_out, &leap_flag,	xl_time_get_leap_second_info(&time_id, &ascii_id_out, &leap_flag,	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
	ascii_utc_time_before_leap, ascii_utc_time_after_leap, ierr)	ascii_utc_time_before, ascii_utc_time_after_leap, ierr)	
xl_get_rotation_angles	xl_get_rotation_angles (xs_initial, ys_initial, zs_initial, xs_final, ys_final, zs_final, ang, ierr)	xl_get_rotation_angles (xs_initial, ys_initial, zs_initial, xs_final, ys_final, zs_final, ang, ierr)	Calculation
xl_get_rotated_vectors	xl_get_rotated_vectors (xs_initial, ys_initial, zs_initial, ang, xs_final, ys_final, zs_final, ierr)	xl_get_rotated_vectors (xs_initial, ys_initial, zs_initial, ang, xs_final, ys_final, zs_final, ierr)	Calculation
xl_quaternions_to_vectors	xl_quaternions_to_vectors (quaternions, ux_vec, uy_vec, uz_vec, ierr)	xl_quaternions_to_vectors (quaternions, ux_vec, uy_vec, uz_vec, ierr)	Calculation
xl_vectors_to_quaternions	xl_vectors_to_quaternions (ux_vec, uy_vec, uz_vec, quaternions, ierr)	xl_vectors_to_quaternions (ux_vec, uy_vec, uz_vec, quaternions, ierr)	Calculation
xl_default_sat_init	xl_default_sat_init(&sat_id, conf_file, ierr)	xl_default_sat_init(&sat_id, conf_file, ierr)	Initialization (Called once)

Table 1: explorer_lib: Mapping V2.3 function usage to V3.7.2 functions

3. LIBRARY EXPLORER_ORBIT: EQUIVALENCES BETWEEN V2.3.X AND V3.7.2

Note that the new initialisation routines for propagation/interpolation expect as input an orbit_id. It means that the initialisation of the propagation/interpolation has to be done in two steps:

- Initialisation of the orbit data using different sources (orbit characteristics, orbit state vector or file)
- Initialisation of the propagator or interpolator itself.

Once an orbit_id is initialised it can be used also to perform time to orbit/orbit to time transformations and compute extra orbit information.
Color code:

- In red: Function parameters present in V2.3.X interface that are not part of V3.7.2 interface
- In green: New function parameters present in the interface of V3.7.2
- In grey: List of parameters that remain in V3.7.2
- In orange: Function whose name has been modified in V3.7.2

Routine name (V2.3)	V2.3	V3.7.2	Use
xo_propag_init_def	xo_propag_init_def(&sat_id, &propag_model, &time_ref, &time0, &orbit0, &time_init_mode, &time, &orbit, &drift_mode, &ascmlst_drift, &inclination, &irep, &icyc, &rlong, &ascmlst, &val_time0, &val_time1, ierr)	xo_orbit_init_def(&sat_id, &time_id, &time_ref, &time0, &orbit0, &drift_mode, &ascmlst_drift, &inclination, &irep, &icyc, &rlong, &ascmlst, &val_time0, &val_time1, &orbit_id, ierr) xo_propag_init(&orbit_id, &propag_model, &time_init_mode, &time_ref, &time, &orbit, &val_time0, &val_time1, &propag_id, ierr)	Initialization (Called once)
xo_propag_cart_init	xo_propag_cart_init(&sat_id, &propag_model, &time_ref, &time, pos_ini, vel_ini, &val_time0, &val_time1, ierr)	xo_orbit_cart_init(&sat_id, &time_id, &time_ref, &time, pos_ini, vel_ini, &abs_orbit, &val_time0, &val_time1, &orbit_id, ierr) xo_propag_init(&orbit_id, &propag_model, &time_init_mode, &time_ref, &time, &orbit, &val_time0, &val_time1, &propag_id, ierr)	Initialization (Called once)
xo_propag_init_file	xo_propag_init_file(&sat_id, &propag_model, &n_files, input_files, &time_init_mode, &time_ref, &time0,	xo_orbit_init_file(&sat_id, &time_id, &orbit_mode, &n_files, input_files, &time_init_mode, &time_ref,	Initialization (Called once)

Routine name (V2.3)	V2.3	V3.7.2	Use
	&time1, &orbit0, &orbit1, &val_time0, &val_time1, ierr)	&time0, &time1, &orbit0, &orbit1, &val_time0, &val_time1, &orbit_id, ierr) xo_propag_init(&orbit_id, &propag_model, &time_init_mode, &time_ref, &time, &orbit, &val_time0, &val_time1, &propag_id, ierr)	
xo_propag	xo_propag(&sat_id, &propag_model, &time_ref, &time, pos, vel, acc, ierr)	xo_propag(&propag_id, &propag_model, &time_ref, &time, pos, vel, acc, ierr)	Calculation
xo_propag_extra	xo_propag_extra(&sat_id, &extra_choice, propag_model_out, propag_extra_out, ierr)	xo_propag_extra(&propag_id, &extra_choice, propag_model_out, propag_extra_out, ierr)	Calculation
xo_propag_close	xo_propag_close(&sat_id)	xo_propag_close(&propag_id, ierr)	Closing (Called once)
xo_interpol_init_file	xo_interpol_init_file(&sat_id, &interpol_model, &n_files, input_files, &time_init_mode, &time_ref, &time0, &time1, &orbit0, &orbit1, &val_time0, &val_time1, ierr)	xo_orbit_init_file(&sat_id, &time_id, &orbit_mode, &n_files, input_files, &time_init_mode, &time_ref, &time0, &time1, &orbit0, &orbit1, &val_time0, &val_time1, &orbit_id, ierr) xo_interpol_init(&orbit_id, &interpol_model, &time_ref, &val_time0, &val_time1, &interpol_id, ierr)	Initialization (Called once)
xo_interpol	xo_interpol(&sat_id, &interpol_model, &time_ref, &time, pos, vel, acc, ierr)	xo_interpol(&interpol_id, &interpol_model, &time_ref, &time, pos, vel, acc, ierr)	Calculation
xo_interpol_extra	xo_interpol_extra(&sat_id, &extra_choice, interpol_model_out, interpol_extra_out, ierr)	xo_interpol_extra(&interpol_id, &extra_choice, interpol_model_out, interpol_extra_out, ierr)	Calculation
xo_interpol_close	xo_interpol_close(&sat_id)	xo_interpol_close(&interpol_id, ierr)	Closing (called once)
xo_orbit_to_time	xo_orbit_to_time(&sat_id, orbit_scenario_file, &orbit, &second, &microsec, &time_ref, &time, ierr)	xo_orbit_to_time(&orbit_id, &orbit, &second, &microsec, &time_ref, &time, ierr)	Calculation
xo_time_to_orbit	xo_time_to_orbit(&sat_id, orbit_scenario_file, &time_ref, &time, &orbit, &second, &microsec, ierr)	xo_time_to_orbit(&orbit_id, &time_ref, &time, &orbit, &second, &microsec, ierr)	Calculation
xo_orbit_info_from_abs	xo_orbit_info_from_abs(&sat_id, orbit_scenario_file, &abs_orbit, &rel_orbit, &cycle, &phase, &res_switch, result_vector, ierr)	xo_orbit_rel_from_abs(&orbit_id, &abs_orbit, &rel_orbit, &cycle, &phase, ierr)	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
		xo_orbit_info(&orbit_id, &abs_orbit, result_vector, ierr)	
xo_orbit_info_from_rel	xo_orbit_info_from_rel(&sat_id, orbit_scenario_file, &abs_orbit, &rel_orbit, &cycle, &phase, &res_switch, result_vector, ierr)	xo_orbit_abs_from_rel(&orbit_id, &rel_orbit, &cycle, &abs_orbit, &phase, ierr) xo_orbit_info(&orbit_id, &abs_orbit, result_vector, ierr)	Calculation
xo_orbit_info_from_phase	xo_orbit_info_from_phase(&sat_id, orbit_scenario_file, &abs_orbit, &rel_orbit, &cycle, &phase, &res_switch, result_vector, ierr)	xo_orbit_abs_from_phase(&orbit_id, &phase, &abs_orbit, &rel_orbit, &cycle, ierr) xo_orbit_info(&orbit_id, &abs_orbit, result_vector, ierr)	Calculation
xo_orbit_changes_close	xo_orbit_changes_close(&sat_id, ierr)	xo_orbit_close(&orbit_id, ierr)	Closing (called once)

Table 2: explorer_orbit: Mapping V2.3 function usage to V3.7.2 functions

4. EXAMPLES: OUTLINE OF THE CALLING SEQUENCES OF EXPLORER_ORBIT FUNCTIONS (PROPAGATION AND INTERPOLATION)

V2.3	V3.7.2
	/* Earth Explorer Ids.: They should be set to NULL always before being initialized !!!! */ xl_time_id time_id = {NULL}; xo_orbit_id orbit_id = {NULL}; xo_propag_id propag_id = {NULL};
long status, ierr[XO_ERR_VECTOR_MAX_LENGTH]; long sat_id;	long status, ierr[XO_ERR_VECTOR_MAX_LENGTH]; long sat_id;
long tri_sat_id, tri_orbit_num; double tri_anx_time, tri_orbit_duration; double tri_time[XL_TIME_TRANS_DIM_MAX]; long tri_ierr[XL_NUM_ERR_TIME_REF_INIT];	long tri_sat_id, tri_orbit_num; double tri_anx_time, tri_orbit_duration; double tri_time[XL_TIME_TRANS_DIM_MAX]; long tri_ierr[XL_NUM_ERR_TIME_REF_INIT];
long propag_model; long time_ref; double time0, time, val_time0, val_time1; long irep, icyc, orbit0, orbit; double ascmlst, rlong; long time_init_mode, drift_mode; double ascmlst_drift, inclination;	long time_ref; double time0, val_time0, val_time1; long irep, icyc, orbit0; double ascmlst, rlong; long drift_mode; double ascmlst_drift, inclination; long propag_model, time_init_mode; double time; long orbit;
double pos[3]; double vel[3]; double acc[3];	double pos[3]; double vel[3]; double acc[3];

```
 long extra_choice;
 double propag_model_out[XO_PROPAG_EXTRA_NUM_DEP_ELEMENTS],
propag_extra_out[XO_PROPAG_EXTRA_NUM_INDEP_ELEMENTS];

 long orbit_t, second_t, microsec_t;
 char *orbit_scenario_file;

 status = xl_time_ref_init(&tri_sat_id, tri_time, &tri_orbit_num,
          &tri_anx_time, &tri_orbit_duration, tri_ierr);

status = xo_propag_init_def(&sat_id, &propag_model, &time_ref,
                        &time0, &orbit0, &time_init_mode, &time, &orbit,
                        &drift_mode, &ascmlst_drift, &inclination,
                        &irep, &icyc, &rlong, &ascmlst,
                        &val_time0, &val_time1, ierr);

 status = xo_propag(&sat_id, &propag_model, &time_ref,
                  &time, pos, vel, acc, ierr);

 status = xo_propag_extra(&sat_id, &extra_choice,
                  propag_model_out, propag_extra_out, ierr);

 status = xo_time_to_orbit ( &sat_id, orbit_scenario_file,
```

```
 long extra_choice;
 double propag_model_out[XO_PROPAG_EXTRA_NUM_DEP_ELEMENTS],
propag_extra_out[XO_PROPAG_EXTRA_NUM_INDEP_ELEMENTS];

 long orbit_t, second_t, microsec_t;

 /* Initialise time_id */
 status = xl_time_ref_init(tri_time, &tri_orbit_num, &tri_anx_time,
              &tri_orbit_duration, &time_id, tri_ierr);

 /* Once time_id is created, we can initialize orbit_id */
 /* Note that orbit_id = sat_id + time_id + orbit data */
 status = xo_orbit_init_def(&sat_id, &time_id, &time_ref, &time0,
                        &orbit0, &drift_mode, &ascmlst_drift, &inclination,
                        &irep, &icyc, &rlong, &ascmlst,
                        &val_time0, &val_time1, &orbit_id, ierr);

 /* Now that we have an orbit_id, we can finally initialise propag_id and
use it to propagate */
 status = xo_propag_init(&orbit_id, &propag_model, &time_init_mode,
                        &time_ref, &time, &orbit,
                        &val_time0, &val_time1,
                        &propag_id, ierr);

 status = xo_propag(&propag_id, &propag_model, &time_ref,
                  &time, pos, vel, acc, ierr);

 status = xo_propag_extra(&propag_id, &extra_choice,
                  propag_model_out, propag_extra_out, ierr);

 /* The orbit_id is also necessary for other orbit related computations */
 status = xo_time_to_orbit (&orbit_id, &time_ref, &time, &orbit_t,
```

```
                &time_ref, &time, &orbit_t, &second_t,
                &microsec_t,   ierr);

status = xo_propag_close(&sat_id);
```

```
                &second_t, &microsec_t, ierr);


/* Finally the Ids are closed */
status = xo_propag_close(&propag_id, ierr);

status = xo_orbit_close(&orbit_id, ierr);

status = xl_time_close(&time_id, ierr);
```

```
long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];
long sat_id;

long tri_sat_id, tri_orbit_num;
double tri_anx_time, tri_orbit_duration;
double tri_time[XL_TIME_TRANS_DIM_MAX];
long tri_ierr[XL_NUM_ERR_TIME_REF_INIT];

long interpol_model;
long time_mode, time_ref;
double time0, time1, val_time0, val_time1;
long orbit0, orbit1;
char **orbit_file;


double time;
double pos[3];
double vel[3];
double acc[3];
```

```
/* Earth Explorer Ids.: They should be set to NULL always before being
initialized !!!! */
xl_time_id     time_id     = {NULL};
xo_orbit_id    orbit_id    = {NULL};
xo_interpol_id  interpol_id   = {NULL};

long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];
long sat_id;

long tri_sat_id, tri_orbit_num;
double tri_anx_time, tri_orbit_duration;
double tri_time[XL_TIME_TRANS_DIM_MAX];
long tri_ierr[XL_NUM_ERR_TIME_REF_INIT];

long time_ref, time_mode, orbit_file_mode;
double time0, time1, val_time0, val_time1;
char **input_files;
long  orbit0, orbit1;

long interpol_model;

double time;
double pos[3];
double vel[3];
double acc[3];
```

```
long extra_choice;
doubleinterpol_model_out[XO_INTERPOL_EXTRA_NUM_DEP_ELEMENTS],
propag_extra_out[XO_INTERPOL_EXTRA_NUM_INDEP_ELEMENTS];

long orbit_t, second_t, microsec_t;
char *orbit_scenario_file;

status = xl_time_ref_init(&tri_sat_id, tri_time, &tri_orbit_num,
          &tri_anx_time, &tri_orbit_duration, tri_ierr);

status = xo_interpol_init_file (&sat_id, &interpol_model,  &n_files,
                    orbit_file,  &time_mode, &time_ref, &time0,  &time1,
                    &orbit0, &orbit1,  &val_time0, &val_time1, ierr);

status = xo_interpol(&sat_id, &interpol_model, &time_ref,
                   &time, pos, vel, acc, ierr);

status = xo_interpol_extra(&sat_id, &extra_choice,
                   interpol_model_out, interpol_extra_out, ierr);

status = xo_time_to_orbit ( &sat_id, orbit_scenario_file,
                    &time_ref, &time, &orbit_t, &second_t, &microsec_t,
                    ierr);
```

```
long extra_choice;
doubleinterpol_model_out[XO_INTERPOL_EXTRA_NUM_DEP_ELEMENTS],
propag_extra_out[XO_INTERPOL_EXTRA_NUM_INDEP_ELEMENTS];

long orbit_t, second_t, microsec_t;

/* Initialise time_id */
status = xl_time_ref_init(tri_time, &tri_orbit_num, &tri_anx_time,
              &tri_orbit_duration, &time_id, tri_ierr);

/* Note that orbit_id = sat_id + time_id + orbit data */
status = xo_orbit_init_file (&sat_id, &time_id, &orbit_file_mode, &n_files,
                    input_files,  &time_mode, &time_ref,  &time0, &time1,
                    &orbit0, &orbit1,  &val_time0, &val_time1,  &orbit_id,
                    ierr);

/* Now that we have an orbit_id, we can finally initialise interpol_id and
use it to interpolate */
status = xo_interpol_init (&orbit_id, &interpol_model,  &time_mode,
                   &time_ref, &val_time0, &val_time1, &interpol_id, ierr);

status =xo_interpol(&interpol_id, &interpol_model, &time_ref,
                   &time,  pos, vel, acc, ierr);

status = xo_interpol_extra(&interpol_id, &extra_choice,
                   interpol_model_out, interpol_extra_out, ierr);

/* The orbit_id is also necessary for other orbit related computations */
status = xo_time_to_orbit (&orbit_id, &time_ref, &time, &orbit_t,
                   &second_t, &microsec_t, ierr);
```

status = xo_interpol_close(&sat_id);	/* Finally the Ids are closed */
	status = xo_interpol_close(&interpol_id, ierr);
	
	status = xo_orbit_close(&orbit_id, ierr);
	
	status = xl_time_close(&time_id, ierr);

Table 3: Calling sequence examples: explorer_orbit routines (propagation and interpolation)

5. LIBRARY EXPLORER_POINTING: EQUIVALENCES BETWEEN V2.3.X AND V3.7.2

The design of the explorer_pointing CFI library in V3.7.2 has been modified in order to support the different attitude modes present in the Earth Explorer Missions. This explains the changes that have been introduced in the attitude functions.
The equivalences between the Attitude Frames used in V2.3 and the ones used in V3.7.2 are as follows:
- SR Frame corresponds to Sat Ref Orbital Frame
- SRR Frame corresponds to Sat Nominal Att Frame
- SRAR Frame corresponds to Sat Attitude Frame

About the xp_target_XXX routines, the spacecraft state vector (time, position, velocity and acceleration) is not part of the interface, as this data (as well as the info associated to the attitude frames that have been initialized and how) is kept into the attitude_id. Note also that in V2.3 a results vector was given as output of the different xp_target_XXX modes. However in V3.7.2 the computation of the target position, velocity, etc, has been moved to another 'extra' routine, called xp_target_extra_vector. Furthermore in the xp_target_extra_XXX functions, the set of output parameters is split into three vectors (no derivative, first derivative and second derivative).
Color code:

- In red: Function parameters present in V2.3.X interface that are not part of V3.7.2 interface
- In green: New function parameters present in the interface of V3.7.2
- In grey: List of parameters that remain in V3.7.2
- In orange: Function whose name has been modified in V3.7.2

Routine name (V2.3)	V2.3	V3.7.2	Use
xp_att_init_def	xp_att_init_def(&sat_id, &aocs_mode, aocs, ierr)	If aocs_mode = XP_AOCS_DEFAULT/XP_AOCS_USER/XP_AOCS_GENERIC: xp_sat_nominal_att_init_model(&model_enum, model_param, &sat_nom_trans_id, ierr) If aocs_mode = XP_AOCS_GPM/XP_AOCS_LNP/XP_AOCS_YSM: xp_sat_nominal_att_init(&aocs_mode, &sat_nom_trans_id, ierr)	Initialization (Called once)
xp_att_init_file (Currently not implemented)	xp_att_init_file(&sat_id, &n_files, attitude_file, &time_init_mode, &time_ref, &time0, &time1, &orbit0, &orbit1, ierr)	xp_sat_nominal_att_init_file(&time_id, &n_files, attitude_file, &time_init_mode, &time_ref, &time0, &time1, &val_time0, &val_time1, &sat_nom_trans_id, ierr)	Initialization (Called once)

Routine name (V2.3)	V2.3	V3.7.2	Use
xp_att_close	xp_att_close(&sat_id, ierr)	xp_sat_nominal_att_close(&sat_nom_trans_id, ierr)	Closing (Called once)
xp_misp_angle_init_def	xp_misp_angle_init_def(&sat_id, &instrument_id, att, att_rate, ierr)	xp_sat_att_angle_init(att, &sat_trans_id, ierr)	Initialization (Called once)
xp_misp_matrix_init_def (Currently not implemented)	xp_misp_matrix_init_def(&sat_id, &instrument_id, att_matrix, att_matrix_rate, ierr)	xp_sat_att_matrix_init_def(att_matrix, &sat_trans_id, ierr)	Initialization (Called once)
xp_misp_bias_init_def	xp_misp_bias_init_def(&sat_id, &instrument_id, bias, sine, cosine, &nodal_period, ierr)	xp_sat_att_init_harmonic(&angle_type, num_terms, harmonic_type_pitch, harmonic_type_roll, harmonic_type_yaw, harmonic_coef_pitch, harmonic_coef_roll, harmonic_coef_yaw, &sat_trans_id, ierr)	Initialization (Called once)
xp_misp_init_file	xp_misp_init_file(&sat_id, &n_files, attitude_file, &time_init_mode, &time_ref, &time0, &time1, &orbit0, &orbit1, ierr)	xp_sat_att_init_file(&time_id, &n_files, attitude_file, &auxiliary_file, time_init_mode, &time_ref, &time0, &time1, &val_time0, &val_time1, &sat_trans_id, ierr)	Initialization (Called once)
xp_misp_close	xp_misp_close(&sat_id, ierr)	xp_sat_att_close(&sat_trans_id, ierr)	Closing (Called once)
xp_attitude and xp_attitude_extra	xp_attitude (&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, ang, ang_rate, ierr) xp_attitude_extra (&sat_id, &instrument_id, model_out, extra_out, ierr)	xp_attitude_init(&attitude_id, ierr) xp_attitude_compute(&time_id, &sat_nom_trans_id, &sat_trans_id, &instr_trans_id, &attitude_id, &time_ref, &time, pos, vel, acc, &target_frame, ierr) Calls to xp_change_frame with the input vector directions {(1,0,0), (0,1,0),(0,0,1)} to get the tranformation matrix from frame_in to frame_out (attitude matrix). xp_change_frame (&sat_id, &time_id, &sat_nom_trans_id, &sat_trans_id, &instr_trans_id, &mode_flag, &frame_flag_in, &frame id_in, &frame_flag_out, &frame_id_out, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, vec_in, vec_rate_in, vec_rate_rate_in, vec_out, vec_rate_out,	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
		vec_rate_rate_out, ierr) Call xl_get_rotation_angles using the three output vectors of xp_change_frame.	
xp_atmos_init	xp_atmos_init(&sat_id, &atmos_mode, &atmos_model, atmos_file, ierr)	xp_atmos_init(&atmos_mode, &atmos_model, atmos_file, &atmos_id, ierr)	Initialization (Called once)
xp_target_inter	xp_target_inter(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &inter_flag, &los_az, &los_el, &geod_alt, &los_az_rate, &los_el_rate, &iray, &freq, target_out, ierr)	xp_target_inter(&sat_id, &attitude_id, &atmos_id, &dem_id, &deriv, &inter_flag, &los_az, &los_el, &geod_alt, &los_az_rate, &los_el_rate, &iray, &freq, &num_user_target, &num_los_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation
xp_target_ground_range	xp_target_ground_range(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &los_az, &los_el, &geod_alt, &distance, &los_az_rate, &los_el_rate, target_out, ierr)	xp_target_ground_range(&sat_id, &attitude_id, &dem_id, &deriv, &los_az, &los_el, &geod_alt, &distance, &los_az_rate, &los_el_rate, &num_user_target, &num_los_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation
xp_target_incidence_angle	xp_target_incidence_angle(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &los_az, &inc_angle, &geod_alt, &los_az_rate, target_out, ierr)	xp_target_incidence_angle(&sat_id, &attitude_id, &dem_id, &deriv, &los_az, &inc_angle, &geod_alt, &los_az_rate, &num_user_target, &num_los_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation
xp_target_range	xp_target_range(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &los_az, &range, &geod_alt, &los_az_rate, &range_rate,	xp_target_range(&sat_id, &attitude_id, &dem_id, &deriv, &los_az, &range, &geod_alt, &los_az_rate, &range_rate, &num_user_target,&num_los_target,	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
	target_out, ierr)	&target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	
xp_target_range_rate	xp_target_range_rate(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &ef_range_rate, &range, &geod_alt, &ef_range_rate_rate, &range_rate, target_out, ierr)	xp_target_range_rate(&sat_id, &attitude_id, &dem_id, &deriv, &ef_range_rate, &range, &geod_alt, &ef_range_rate_rate, &range_rate, &num_user_target, &num_los_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation
xp_target_tangent	xp_target_tangent(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &los_az, &los_el, &los_az_rate, &los_el_rate, &iray, &freq, target_out, ierr)	xp_target_tangent(&sat_id, &attitude_id, &atmos_id, &dem_id, &deriv, &los_az, &los_el, &los_az_rate, &los_el_rate, &iray, &freq, &num_user_target, &num_los_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation
xp_target_altitude	xp_target_altitude(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &los_az, &geod_alt, &los_az_rate, &iray, &freq, target_out, ierr)	xp_target_altitude(&sat_id, &attitude_id, &atmos_id, &dem_id, &deriv, &los_az, &geod_alt, &los_az_rate, &iray, &freq, &num_user_target, &num_los_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation
xp_target_star	xp_target_star(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &star_ra, &star_dec, &star_ra_rate, &star_dec_rate, &iray, &freq, target_out, ierr)	xp_target_star(&sat_id, &attitude_id, &atmos_id, &dem_id, &deriv, &star_ra, star_dec, &&star_ra_rate, &star_dec_rate, &iray, &freq, &num_user_target, &num_los_target, &target_id, ierr)	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
		xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	
xp_target_station	xp_target_station(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &geoc_long, &geod_lat, &geod_alt, &min_link_el, target_out, ierr)	xp_target_station(&sat_id, &attitude_id, &dem_id, &deriv, &geoc_long, &geod_lat, &geod_alt, &min_link_el, &num_user_target, &num_los_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation
xp_target_drs	xp_target_drs(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, drs_pos, drs_vel, target_out, ierr)	xp_target_drs(&sat_id, &attitude_id, &dem_id, &deriv, drs_pos, drs_vel, &num_user_target, &num_los_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation
xp_target_generic	xp_target_generic(&sat_id, &instrument_id, &time_ref, &time, &deriv, targ_pos, targ_vel, target_out, ierr)	xp_target_generic(&sat_id, &attitude_id, &dem_id, &deriv, targ_pos, targ_vel, targ_acc, &num_user_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation
xp_target_extra_main	xp_target_extra_main(&sat_id, &choice, main_extra_results, ierr)	xp_target_extra_main (&target_id, &choice, &target_type, &target_number, main_results, main_results_rate, main_results_rate_rate, ierr)	Calculation
xp_target_extra_aux	xp_target_extra_aux(&sat_id, &choice, extra_results, ierr)	xp_target_extra_aux(&target_id, &choice, &target_type, &target_number, aux_results, aux_results_rate, aux_results_rate_rate, ierr)	Calculation
xp_target_extra_ef_target	xp_target_extra_ef_target(&sat_id, &freq, extra_results, ierr)	xp_target_extra_ef_target(&target_id, &choice, &target_type, &target_number, &freq,	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
		ef_target_results_rate, ef_target_results_rate_rate, ierr)	
xp_target_extra_target_to_sun	xp_target_extra_target_to_sun (&sat_id, &deriv_flag, &iray, &freq, extra_results, ierr)	xp_target_extra_target_to_sun (&target_id, &choice, &target_type, &target_number, &iray, &freq, sun_results, sun_results_rate, sun_results_rate_rate, ierr)	Calculation
xp_sun	xp_sun(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &iray, &freq, target_out, ierr)	xp_target_tangent_sun(&sat_id, &attitude_id, &atmos_id, &dem_id, &deriv, &iray, &freq, &num_user_target, &num_los_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation
xp_moon	xp_moon(&sat_id, &instrument_id, &time_ref, &time, pos, vel, acc, &deriv, &iray, &freq, target_out, ierr)	xp_target_tangent_moon(&sat_id, &attitude_id, &atmos_id, &dem_id, &deriv, &iray, &freq, &num_user_target, &num_los_target, &target_id, ierr) xp_target_extra_vector (&target_id, &choice, &target_type, &target_number, vector_results, vector_results_rate, vector_results_rate_rate, ierr)	Calculation

Table 4: explorer_pointing: Mapping V2.3 function usage to V3.7.2 functions

6. EXAMPLE: OUTLINE OF CALLING SEQUENCES OF EXPLORER_POINTING FUNCTIONS (GEOLOCATION)

V2.3	V3.7.2
	/* Earth Explorer Ids.: They should be set to NULL always before being initialized !!!! */ xl_time_id time_id = {NULL}; xp_sat_nom_trans_id sat_nom_trans_id = {NULL}; xp_sat_trans_id sat_trans_id = {NULL}; xp_instr_trans_id instr_trans_id = {NULL}; xp_attitude_id attitude_id = {NULL};

```
                                                xp_atmos_id  atmos_id = {NULL};
                                                xp_dem_id  dem_id = {NULL};
                                                xp_target_id  target_id = {NULL};

long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];    long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];
long sat_id;                                    long sat_id;

long tri_sat_id, tri_orbit_num;                 long tri_sat_id, tri_orbit_num;
double tri_anx_time, tri_orbit_duration;        double tri_anx_time, tri_orbit_duration;
double tri_time[XL_TIME_TRANS_DIM_MAX];         double tri_time[XL_TIME_TRANS_DIM_MAX];
long tri_ierr[XL_NUM_ERR_TIME_REF_INIT];        long tri_ierr[XL_NUM_ERR_TIME_REF_INIT];

long aocs_mode;                                 long aocs_mode;
double aocs[XP_NUM_MODEL_PARAM];

long n_files, time_init_mode, time_ref, orbit0, orbit1;
char **mispointing_file;
double time0, time1;

                                                long n_files, time_init_mode, time_ref;
                                                char **attitude_file *auxiliary_file;
                                                double time0, time1, val_time0, val_time1;

                                                double ang[3], offset[3];

long atmos_mode, atmos_model;                   long atmos_mode, atmos_model;
char atmos_file[XL_MAX_STR];                    char atmos_file[XL_MAX_STR];

                                                long target_frame;
                                                double time, pos[3], vel[3], acc[3];

long  deriv, inter_flag, iray, instrument_id;   long deriv;
double time, pos[3], vel[3], acc[3];            long inter_flag, iray;
double los_az, los_el, geod_alt, los_az_rate, los_el_rate, freq;
                                                double los_az, los_el, geod_alt, los_az_rate, los_el_rate, freq;
double target_out[14];                          long num_user_target, num_los_target;

                                                long choice, target_type, target_number;
                                                double vector_results[XP_SIZE_TARGET_RESULT_VECTOR],
                                                vector_results_rate[XP_SIZE_TARGET_RESULT_VECTOR],
```

```
long choice;
double main_extra_results[XP_TARGET_EXTRA_MAIN_OUTPUT_SIZE];
```

```
status = xl_time_ref_init(&tri_sat_id, tri_time, &tri_orbit_num,
          &tri_anx_time, &tri_orbit_duration, tri_ierr);
```

```
aocs_mode   =  XP_AOCS_YSM;
status = xp_att_init_def(&sat_id, &aocs_mode, aocs, ierr);
```

```
status = xp_misp_init_file(&sat_id, &n_files, mispointing_file,
                    &time_init_mode, &time_ref, &time0, &time1,
                    &orbit0, &orbit1,
                    ierr);
```

```
vector_results_rate_rate[XP_SIZE_TARGET_RESULT_VECTOR];

 double main_results[XP_SIZE_TARGET_RESULT_MAIN],
main_results_rate[XP_SIZE_TARGET_RESULT_MAIN],
main_results_rate_rate[XP_SIZE_TARGET_RESULT_MAIN];


  /* Initialise first all the Ids that we will need to initialize afterwards the
     attitude_id */

  /* Initialise time_id */
  status = xl_time_ref_init(tri_time, &tri_orbit_num, &tri_anx_time,
                &tri_orbit_duration, &time_id, tri_ierr);

  /* Initialise sat_nom_trans_id */
  aocs_mode   =  XP_AOCS_YSM;
  status = xp_sat_nominal_att_init(&aocs_mode, &sat_nom_trans_id, ierr);

  /* Initialise sat_trans_id */
  status = xp_sat_att_init_file(&time_id, &n_files, attitude_file,
                    &auxiliary_file, time_init_mode, &time_ref, &time0,
                    &time1, &val_time0, &val_time1, &sat_trans_id, ierr);

   /* Initialise instr_trans_id */
  status = xp_instr_att_angle_init(ang, offset, &instr_trans_id, ierr);

  /* In order to get an attitude_id, it is necessary to call the following two
routines sequentially  (attitude_id is an input/output parameter in
xp_attitude_compute)*/
  status = xp_attitude_init(&attitude_id, ierr);

  status = xp_attitude_compute(&time_id,   &sat_nom_trans_id,
                    &sat_trans_id,   &instr_trans_id,   &attitude_id,
                    &time_ref, &time, pos, vel, acc,    &target_frame,   ierr);
```

```
status = xp_atmos_init(&sat_id, &atmos_mode, &atmos_model, atmos_file,
ierr);

status = xp_target_inter(&sat_id, &instrument_id, &time_ref,  &time, pos,
                         vel, acc, &deriv, &inter_flag, &los_az,  &los_el,
                         &geod_alt, &los_az_rate, &los_el_rate, &iray,
                         &freq, target_out, ierr);

status = xp_target_extra_main(&sat_id, &choice, main_extra_results,
                              ierr);
```

```
/* Now that we have an attitude_id, we can also perform geo-location
   calculations . xp_target_XXX generates a target_id that is used
   afterwards in the xp_target_extra functions to compute the selected
   results. For this particular example we will consider atmosphere (calling
   xp_atmos_init) and we will not make use of the DEM (not calling
   xp_dem_init)*/

status = xp_atmos_init(&atmos_mode, &atmos_model, atmos_file,
                       &atmos_id, ierr);

status = xp_target_inter(&sat_id,  &attitude_id,  &atmos_id,  &dem_id,
         &deriv, &inter_flag, &los_az, &los_el, &geod_alt,  &los_az_rate,
         &los_el_rate, &iray, &freq,  &num_user_target,
&num_los_target,
         &target_id, ierr);

status = xp_target_extra_vector (&target_id, &choice,  &target_type,
         &target_number,  vector_results,  vector_results_rate,
         vector_results_rate_rate, ierr);

status = xp_target_extra_main (&target_id, &choice, &target_type,
         &target_number,  main_results, main_results_rate,
         main_results_rate_rate, ierr);

/* Finally the Ids are closed */

status = xp_target_close(&target_id, ierr);
```

status = xp_misp_close(&sat_id, ierr);
status = xp_att_close(&sat_id, ierr);
	status = xp_attitude_close(&attitude_id, ierr);
	status = xp_atmos_close(&atmos_id, ierr);
	status = xp_instr_att_close(&instr_trans_id, ierr);
	status = xp_sat_att_close(&sat_trans_id, ierr);
	status = xp_sat_nominal_att_close(&sat_nom_trans_id, ierr);
	status = xl_time_close(&time_id, ierr);

Table 5: Calling sequence examples: explorer_pointing routines (geolocation)

7. LIBRARY EXPLORER_VISIBILITY: EQUIVALENCES BETWEEN V2.3.X AND V3.7.2

The visibility routines in V3.7.2 benefit from the use of the Ids, particularly of the orbit_id, such that the orbit scenario file is no longer part of the interface.
Then, once an orbit_id is initialised it can be used perform any kind of calculation that generates visibility segments.
Color code:

- In red: Function parameters present in V2.3.X interface that are not part of V3.7.2 interface
- In green: New function parameters present in the interface of V3.7.2
- In grey: List of parameters that remain in V3.7.2
- In orange: Function whose name has been modified in V3.7.2

Routine name (V2.3)	V2.3	V3.7.2	Use
xv_zone_vis_time	xv_zone_vis_time(&sat_id, orbit_scenario_file, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, swath_file, zone_id, zone_db_file, &projection, &zone_num, zone_long, zone_lat, &zone_diam, &min_duration, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &end_orbit, &end_second, &end_microsec, &end_cycle, &coverage,ierr)	xv_zone_vis_time(&orbit_id, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, &swath_flag, swath_file, zone_id, zone_db_file, &projection, &zone_num, zone_long, zone_lat, &zone_diam, &min_duration, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &end_orbit, &end_second, &end_microsec, &end_cycle, &coverage,ierr)	Calculation
xv_station_vis_time	xv_station_vis_time(&sat_id, orbit_scenario_file, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, &swath_file, sta_id, sta_db_file, &mask, &aos_elevation, &los_elevation, &min_duration, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &end_orbit, &end_second, &end_microsec, &end_cycle, &zdop_orbit, zdop_second, &zdop_ierr)	xv_station_vis_time(&orbit_id, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, &swath_flag, &swath_file, sta_id, sta_db_file, &mask, &aos_elevation, &los_elevation, &min_duration, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &end_orbit, &end_second, &end_microsec, &end_cycle, &zdop_orbit, &zdop_second, &zdop_microsec, &zdop_cycle, ierr)	Calculation
xv_drs_vis_time	xv_drs_vis_time(&sat_id, orbit_scenario_file, &orbit_type, &start_orbit, &start_cycle,	xv_drs_vis_time(&orbit_id, &sat_nom_trans_id, &sat_trans_id, &instr_trans_id, &orbit_type,	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
	&stop_orbit, &stop_cycle, &longitude, &min_duration, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &end_orbit, &end_second, &end_microsec, &end_cycle, ierr)	&start_orbit, &start_cycle, &stop_orbit, &stop_cycle, &longitude, &min_duration, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &end_orbit, &end_second, &end_microsec, &end_cycle, ierr)	
xv_swath_pos	xv_swath_pos(&sat_id, orbit_scenario_file, swath_file, &orbit_type, &orbit, &second, &microsec, &cycle, longitude, latitude, altitude, ierr)	xd_read_stf (swath_file, &stf_data, ierr) xv_swath_pos(&orbit_id, &stf_data, &orbit_type, &orbit, &second, &microsec, &cycle, longitude, latitude, altitude, ierr)	Calculation
xv_star_vis_time	xv_star_vis_time(&sat_id, orbit_scenario_file, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, swath_file_upper, swath_file_lower, star_id, star_db_file, &star_ra, &star_dec, &min_duration, &star_ra_deg, &star_dec_deg, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &bgn_coverage, &end_orbit, &end_second, &end_microsec, &end_cycle, &end_coverage, ierr)	xv_star_vis_time(&orbit_id, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, &swath_flag, swath_file_upper, swath_file_lower, star_id, star_db_file, &star_ra, &star_dec, &min_duration, &star_ra_deg, &star_dec_deg, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &bgn_coverage, &end_orbit, &end_second, &end_microsec, &end_cycle, &end_coverage, ierr)	Calculation
xv_multizones_vis_time	xv_multizones_vis_time(&sat_id, orbit_scenario_file, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, swath_file, &num_zones, zone_id, zone_db_file, projection, zone_num, zone_long, zone_lat, zone_diam, &min_duration, &extra_info_flag, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &end_orbit, &end_second, &end_microsec, &end_cycle, &nb_zon_in_segment, &zones_in_segment, &coverage, ierr)	xv_multizones_vis_time(&orbit_id, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, &swath_flag, swath_file, &num_zones, zone_id, zone_db_file, projection, zone_num, zone_long, zone_lat, zone_diam, &min_duration, &extra_info_flag, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &end_orbit, &end_second, &end_microsec, &end_cycle, &nb_zon_in_segment, &zones_in_segment, &coverage, ierr)	Calculation
xv_multistations_vis_time	xv_multistations_vis_time(&sat_id, orbit_scenario_file, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, swath_file, &num_stations, station_db_file, station_id,	xv_multistations_vis_time(&orbit_id, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, &swath_flag, swath_file, &num_stations, station_db_file, station_id, aos_elevation,	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
	aos_elevation, los_elevation, mask, &min_duration, &extra_info_flag, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &end_orbit, &end_second, &end_microsec, &end_cycle, &zdop_orbit, &zdop_second, &zdop_microsec, &zdop_cycle, &nb_stat_in_segment, &stat_in_segment, ierr)	los_elevation, mask, &min_duration, &extra_info_flag, &number_segments, &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle, &end_orbit, &end_second, &end_microsec, &end_cycle, &zdop_orbit, &zdop_second, &zdop_microsec, &zdop_cycle, &nb_stat_in_segment, &stat_in_segment, ierr)	
xv_orbit_extra	xv_orbit_extra (&sat_id, &orbit, orbit_info_vector, &num_sza, sza, &sza_up, &sza_down, &eclipse_entry, &eclipse_exit, &sun_moon_entry, &sun_moon_exit, ierr)	xv_orbit_extra (&orbit_id, &orbit, orbit_info_vector, &num_sza, sza, &sza_up, &sza_down, &eclipse_entry, &eclipse_exit, &sun_moon_entry, &sun_moon_exit, ierr)	Calculation
xv_time_segments_not	xv_time_segments_not(&sat_id, orbit_scenario_file, &orbit_type, &order_switch, &number_segments_in, bgn_orbit_in, bgn_secs_in, bgn_microsecs_in, bgn_cycle_in, end_orbit_in, end_secs_in, end_microsecs_in, end_cycle_in, &num_segments_out, &bgn_orbit_out, &bgn_secs_out, &bgn_microsecs_out, &bgn_cycle_out, &end_orbit_out, &end_secs_out, &end_microsecs_out, &end_cycle_out, ierr)	xv_time_segments_not(&orbit_id, &orbit_type, &order_switch, &number_segments_in, bgn_orbit_in, bgn_secs_in, bgn_microsecs_in, bgn_cycle_in, end_orbit_in, end_secs_in, end_microsecs_in, end_cycle_in, &num_segments_out, &bgn_orbit_out, &bgn_secs_out, &bgn_microsecs_out, &bgn_cycle_out, &end_orbit_out, &end_secs_out, &end_microsecs_out, &end_cycle_out, ierr)	Calculation
xv_time_segments_or	xv_time_segments_or (&sat_id, orbit_scenario_file, &orbit_type, &order_switch, &number_segments_1, bgn_orbit_1, bgn_second_1, bgn_microsec_1, bgn_cycle_1, end_orbit_1, end_second_1, end_microsec_1, end_cycle_1, &number_segments_2, bgn_orbit_2, bgn_second_2, bgn_microsec_2, bgn_cycle_2, end_orbit_2, end_second_2, end_microsec_2, end_cycle_2, &num_segments_out, &bgn_orbit_out, &bgn_secs_out, &bgn_microsecs_out, &bgn_cycle_out, &end_orbit_out, &end_secs_out, &end_microsecs_out, &end_cycle_out, ierr)	xv_time_segments_or (&orbit_id, &orbit_type, &order_switch, &number_segments_1, bgn_orbit_1, bgn_second_1, bgn_microsec_1, bgn_cycle_1, end_orbit_1, end_second_1, end_microsec_1, end_cycle_1, &number_segments_2, bgn_orbit_2, bgn_second_2, bgn_microsec_2, bgn_cycle_2, end_orbit_2, end_second_2, end_microsec_2, end_cycle_2, &num_segments_out, &bgn_orbit_out, &bgn_secs_out, &bgn_microsecs_out, &bgn_cycle_out, &end_orbit_out, &end_secs_out, &end_microsecs_out, &end_cycle_out, ierr)	Calculation
xv_time_segments_and	xv_time_segments_and (&sat_id, orbit_scenario_file,	xv_time_segments_and (&orbit_id, &orbit_type,	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
	&orbit_type, &order_switch, &number_segments_1, bgn_orbit_1, bgn_second_1, bgn_microsec_1, bgn_cycle_1, end_orbit_1, end_second_1, end_microsec_1, end_cycle_1, &number_segments_2, bgn_orbit_2, bgn_second_2, bgn_microsec_2, bgn_cycle_2, end_orbit_2, end_second_2, end_microsec_2, end_cycle_2, &num_segments_out, &bgn_orbit_out, &bgn_secs_out, &bgn_microsecs_out, &bgn_cycle_out, &end_orbit_out, &end_secs_out, &end_microsecs_out, &end_cycle_out, ierr)	&order_switch, &number_segments_1, bgn_orbit_1, bgn_second_1, bgn_microsec_1, bgn_cycle_1, end_orbit_1, end_second_1, end_microsec_1, end_cycle_1, &number_segments_2, bgn_orbit_2, bgn_second_2, bgn_microsec_2, bgn_cycle_2, end_orbit_2, end_second_2, end_microsec_2, end_cycle_2, &num_segments_out, &bgn_orbit_out, &bgn_secs_out, &bgn_microsecs_out, &bgn_cycle_out, &end_orbit_out, &end_secs_out, &end_microsecs_out, &end_cycle_out, ierr)	
xv_time_segments_sort	xv_time_segments_sort (&sat_id, orbit_scenario_file, &orbit_type, &sort_criteria, &number_segments, bgn_orbit, bgn_second, bgn_microsec, bgn_cycle, end_orbit, end_second, end_microsec, end_cycle, ierr)	xv_time_segments_sort (&orbit_id, &orbit_type, &sort_criteria, &number_segments, bgn_orbit, bgn_second, bgn_microsec, bgn_cycle, end_orbit, end_second, end_microsec, end_cycle, ierr)	Calculation
xv_time_segments_merge	xv_time_segments_merge(&sat_id, orbit_scenario_file, &orbit_type, &order_switch, &number_segments, bgn_orbit, bgn_secs, bgn_microsecs, bgn_cycle, end_orbit, end_secs, end_microsecs, end_cycle, &num_segments_out, &bgn_orbit_out, &bgn_secs_out, &bgn_microsecs_out, &bgn_cycle_out, &end_orbit_out, &end_secs_out, &end_microsecs_out, &end_cycle_out, ierr)	xv_time_segments_merge(&orbit_id, &orbit_type, &order_switch, &number_segments, bgn_orbit, bgn_secs, bgn_microsecs, bgn_cycle, end_orbit, end_secs, end_microsecs, end_cycle, &num_segments_out, &bgn_orbit_out, &bgn_secs_out, &bgn_microsecs_out, &bgn_cycle_out, &end_orbit_out, &end_secs_out, &end_microsecs_out, &end_cycle_out, ierr)	Calculation
xv_time_segments_delta	xv_time_segments_delta(&sat_id, orbit_scenario_file, &operation, &orbit_type, &delta_secs, &delta_microsecs, &number_segments, bgn_orbit, bgn_secs, bgn_microsecs, bgn_cycle, end_orbit, end_secs, end_microsecs, end_cycle, &num_segments_out, &bgn_orbit_out, &bgn_secs_out, &bgn_microsecs_out, &bgn_cycle_out, &end_orbit_out, &end_secs_out,	xv_time_segments_delta(&orbit_id, &orbit_type, &entry_offset, &exit_offset, &number_segments, bgn_orbit, bgn_secs, bgn_microsecs, bgn_cycle, end_orbit, end_secs, end_microsecs, end_cycle, &num_segments_out, &bgn_orbit_out, &bgn_secs_out, &bgn_microsecs_out, &bgn_cycle_out, &end_orbit_out, &end_secs_out, &end_microsecs_out, &end_cycle_out, ierr)	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
	&end_microsecs_out, &end_cycle_out, ierr)		
xv_time_segments_mapping	xv_time_segments_mapping(&sat_id, orbit_scenario_file, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, swath_file, &zone_num, zone_id, zone_db_file, &projection, &zone_diam, zone_long, zone_lat, &num_mappings,&num_segments, &orbit_direction, &bgn_orbit, &bgn_secs,&bgn_microsec, &bgn_cycle, &end_orbit, &end_secs,&end_microsec, &end_cycle, &coverage,ierr)	xv_time_segments_mapping(&orbit_id, &orbit_type, &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, &swath_flag, swath_file, &zone_num, zone_id, zone_db_file, &projection, &zone_diam, zone_long, zone_lat, &num_mappings,&num_segments, &orbit_direction, &bgn_orbit, &bgn_secs,&bgn_microsec, &bgn_cycle, &end_orbit, &end_secs,&end_microsec, &end_cycle, &coverage,ierr)	Calculation

Table 6: explorer_visibility: Mapping V2.3 function usage to V3.7.2 functions

8. EXAMPLE: OUTLINE OF THE CALLING SEQUENCE OF AN EXPLORER_VISIBILITY FUNCTION (VISIBILITY SEGMENTS COMPUTATION)

V2.3	V3.7.2
	/* Earth Explorer Ids.: They should be set to NULL always before being initialized !!!! */ xl_time_id time_id = {NULL}; xo_orbit_id orbit_id = {NULL}; xo_propag_id propag_id = {NULL};
long status, ierr[XO_ERR_VECTOR_MAX_LENGTH]; long sat_id; long tri_sat_id, tri_orbit_num; double tri_anx_time, tri_orbit_duration; double tri_time[XL_TIME_TRANS_DIM_MAX]; long tri_ierr[XL_NUM_ERR_TIME_REF_INIT];	long status, ierr[XO_ERR_VECTOR_MAX_LENGTH]; long sat_id; long tri_sat_id, tri_orbit_num; double tri_anx_time, tri_orbit_duration; double tri_time[XL_TIME_TRANS_DIM_MAX]; long tri_ierr[XL_NUM_ERR_TIME_REF_INIT];
long propag_model; long time_ref; double time0, time, val_time0, val_time1; long irep, icyc, orbit0, orbit; double ascmlst, rlong; long time_init_mode, drift_mode; double ascmlst_drift, inclination;	long time_ref; double time0, val_time0, val_time1; long irep, icyc, orbit0; double ascmlst, rlong; long drift_mode; double ascmlst_drift, inclination;
long orbit_type, start_orbit, start_cycle, stop_orbit, stop_cycle, zone_num, projection, number_segments, *bgn_orbit, *bgn_second, *bgn_microsec, *bgn_cycle, *end_orbit, *end_second, *end_microsec, *end_cycle, *coverage;	long swath_flag, orbit_type, start_orbit, start_cycle, stop_orbit, stop_cycle, zone_num, projection, number_segments, *bgn_orbit, *bgn_second, *bgn_microsec, *bgn_cycle, *end_orbit, *end_second, *end_microsec, *end_cycle, *coverage;

```
double *zone_long, *zone_lat, zone_diam, min_duration;
char *orbit_scenario_file, *swath_file;
char zone_id[8], *zone_db_file;


status = xl_time_ref_init(&tri_sat_id, tri_time, &tri_orbit_num,
          &tri_anx_time, &tri_orbit_duration, tri_ierr);









status = xv_zone_vis_time(&sat_id, orbit_scenario_file,  &orbit_type,
      &start_orbit, &start_cycle, &stop_orbit, &stop_cycle, swath_file,
      zone_id, zone_db_file, &projection, &zone_num, zone_long,
      zone_lat, &zone_diam, &min_duration, &number_segments,
      &bgn_orbit, &bgn_second, &bgn_microsec, &bgn_cycle,
      &end_orbit, &end_second, &end_microsec, &end_cycle,
      &coverage,ierr)
```

```
double *zone_long, *zone_lat, zone_diam, min_duration;
char *swath_file;
char zone_id[8], *zone_db_file;

/* Initialise time_id */
status = xl_time_ref_init(tri_time, &tri_orbit_num, &tri_anx_time,
          &tri_orbit_duration, &time_id, tri_ierr);

/* Once time_id is created, we can initialize orbit_id */
/* Note that orbit_id = sat_id + time_id + orbit data */
status = xo_orbit_init_def(&sat_id, &time_id, &time_ref, &time0,
                  &orbit0, &drift_mode, &ascmlst_drift, &inclination,
                  &irep, &icyc, &rlong, &ascmlst,
                  &val_time0, &val_time1, &orbit_id, ierr);

/* Now that we have an orbit_id, we can finally perform a visibility
segments calculation */
status = xv_zone_vis_time(&orbit_id, &orbit_type, &start_orbit,
          &start_cycle, &stop_orbit, &stop_cycle, &swath_flag,
          swath_file, zone_id, zone_db_file, &projection, &zone_num,
          zone_long, zone_lat, &zone_diam, &min_duration,
          &number_segments, &bgn_orbit, &bgn_second,
          &bgn_microsec, &bgn_cycle, &end_orbit, &end_second,
          &end_microsec, &end_cycle, &coverage,ierr);


/* Finally the Ids are closed */

status = xo_orbit_close(&orbit_id, ierr);

status = xl_time_close(&time_id, ierr);
```

Table 7: Calling sequence examples: explorer_visibility routines (visibility segments computation)

9. LIBRARY EXPLORER_GEN_FILES: EQUIVALENCES BETWEEN V2.3.X AND V3.7.2

In V3.7.2 the explorer_gen_files library does not longer exist. The functions belonging to this library have been integrated into explorer_visibility (xv_gen_swath) and explorer_orbit (remaining xo_gen_XXX functions). The new routines also make use of the Ids. As a result we can observe that, for instance, the interface of the function xg_gen_swath has been simplified.
Color code:

- In red: Function parameters present in V2.3.X interface that are not part of V3.7.2 interface
- In green: New function parameters present in the interface of V3.7.2
- In grey: List of parameters that remain in V3.7.2
- In orange: Function whose name has been modified in V3.7.2

Routine name (V2.3)	V2.3	V3.7.2	Use
xg_gen_swath	xg_gen_swath (&sat_id, orbit_scenario_file, &requested_orbit, swath_definition_file, &repeat_cycle, &cycle_length, &drift_mode, &mlst_drift, &inclination, dir_name, instr_swath_file_suffix, file_class, &version_number, fh_system, ierr)	xv_gen_swath (&orbit_id, &atmos_id, &requested_orbit, swath_definition_file, dir_name, swath_file, file_class, &version_number, fh_system, ierr)	Calculation
xg_gen_osf_create	xg_gen_osf_create (&sat_id, &abs_orbit_number, &cycle_number, &phase_number, &repeat_cycle, &cycle_length, &anx_long, &drift_mode, &inclination, &mlst_drift, &mlst, &date, output_dir, output_filename, file_class, &version_number, fh_system, ierr)	xo_gen_osf_create (&sat_id, &time_id, &abs_orbit_number, &cycle_number, &phase_number, &repeat_cycle, &cycle_length, &anx_long, &drift_mode, &inclination, &mlst_drift, &mlst, &date, output_dir, output_filename, file_class, &version_number, fh_system, ierr)	Calculation
xg_gen_osf_append_orbit_change	xg_gen_osf_append_orbit_change (&sat_id, &input_filename, &abs_orbit_number, &repeat_cycle, &cycle_length, &anx_long, &drift_mode, &inclination, &mlst_drift, &mlst, &phase_increment, output_dir, output_filename, file_class, &version_number, fh_system, ierr)	xo_gen_osf_append_orbit_change (&sat_id, &time_id, &input_filename, &abs_orbit_number, &repeat_cycle, &cycle_length, &anx_long, &drift_mode, &inclination, &mlst_drift, &mlst, &phase_increment, output_dir, output_filename, file_class, &version_number, fh_system, ierr)	Calculation
xg_gen_osf_change_repeat_cycle	xg_gen_osf_change_repeat_cycle (&sat_id,	xo_gen_osf_change_repeat_cycle (&sat_id,	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
	&input_filename, &abs_orbit_number, &search_direction, &repeat_cycle, &cycle_length, &anx_long, &drift_mode, &inclination, &mlst_drift, &phase_increment, output_dir, output_filename, file_class, &version_number, fh_system, ierr)	&time_id, &input_filename, &abs_orbit_number, &search_direction, &repeat_cycle, &cycle_length, &anx_long, &drift_mode, &inclination, &mlst_drift, &phase_increment, output_dir, output_filename, file_class, &version_number, fh_system, ierr)	
xg_gen_osf_add_drift_cycle	xg_gen_osf_add_drift_cycle (&sat_id, &input_filename, &drift_start_orbit, &drift_stop_orbit, &drift_stop_anx_long, &max_altitude_change, &phase_inc_start, &phase_inc_stop, output_dir, output_filename, file_class, &version_number, fh_system, ierr)	xo_gen_osf_add_drift_cycle (&sat_id, &time_id, &input_filename, &drift_start_orbit, &drift_stop_orbit, &drift_stop_anx_long, &max_altitude_change, &phase_inc_start, &phase_inc_stop, output_dir, output_filename, file_class, &version_number, fh_system, ierr)	Calculation
xg_gen_rof	xg_gen_rof(&sat_id, &time_init, &time_ref, &start_time,&stop_time, &start_orbit, &stop_orbit, &osv_interval, &osv_precise, &ref_filetype, reference_file, &rof_filetype, output_dir, rof_filename, file_class, &version_number, fh_system, ierr)	xo_gen_rof(&sat_id, &time_id, &time_init, &time_ref, &start_time,&stop_time, &start_orbit, &stop_orbit, &osv_interval, &osv_precise, &ref_filetype, reference_file, &rof_filetype, output_dir, rof_filename, file_class, &version_number, fh_system, ierr)	Calculation
xg_gen_rof_prototype	xg_gen_rof_prototype (&sat_id, &propag_model, &time_ref, &time0, &orbit0, &time_init_mode, &start_time, &start_orbit &stop_time, &stop_orbit, &drift_mode, &ascmlst_drift, &inclination, &irep, &icyc, &rlong, &ascmlst, &osv_interval output_dir,rof_filename, file_class, &version_number, fh_system, ierr)	xo_gen_rof_prototype (&sat_id, &time_id, &propag_model, &time_ref, &time0, &orbit0, &time_init_mode, &start_time, &start_orbit &stop_time, &stop_orbit, &drift_mode, &ascmlst_drift, &inclination, &irep, &icyc, &rlong, &ascmlst, &osv_interval output_dir,rof_filename, file_class, &version_number, fh_system, ierr)	Calculation
xg_gen_pof	xg_gen_pof(&sat_id, &time_init, &time_ref, &start_time,&stop_time, &start_orbit, &stop_orbit, &osv_location, &ref_filetype, reference_file, &pof_filetype, output_dir,pof_filename, file_class, &version_number, fh_system, ierr)	xo_gen_pof(&sat_id, &time_id, &time_init, &time_ref, &start_time,&stop_time, &start_orbit, &stop_orbit, &osv_location, &ref_filetype, reference_file, &pof_filetype, output_dir,pof_filename, file_class, &version_number, fh_system, ierr)	Calculation
xg_gen_dnf	xg_gen_dnf(&sat_id, &time_init, &time_ref, &start_time,&stop_time, &start_orbit,	xo_gen_dnf(&sat_id, &time_id, &time_init, &time_ref, &start_time, &stop_time, &start_orbit,	Calculation

Routine name (V2.3)	V2.3	V3.7.2	Use
	&stop_orbit, &osv_interval, &osv_precise, &ref_filetype, reference_file, ctrl_file, &dnf_filetype, output_dir, dnf_filename, file_class, &version_number, fh_system, ierr)	&stop_orbit, &osv_interval, &osv_precise, &ref_filetype, reference_file, ctrl_file, &dnf_filetype, output_dir, dnf_filename, file_class, &version_number, fh_system, ierr)	

Table 8: explorer_gen_files: Mapping V2.3 function usage to V3.7.2 functions