

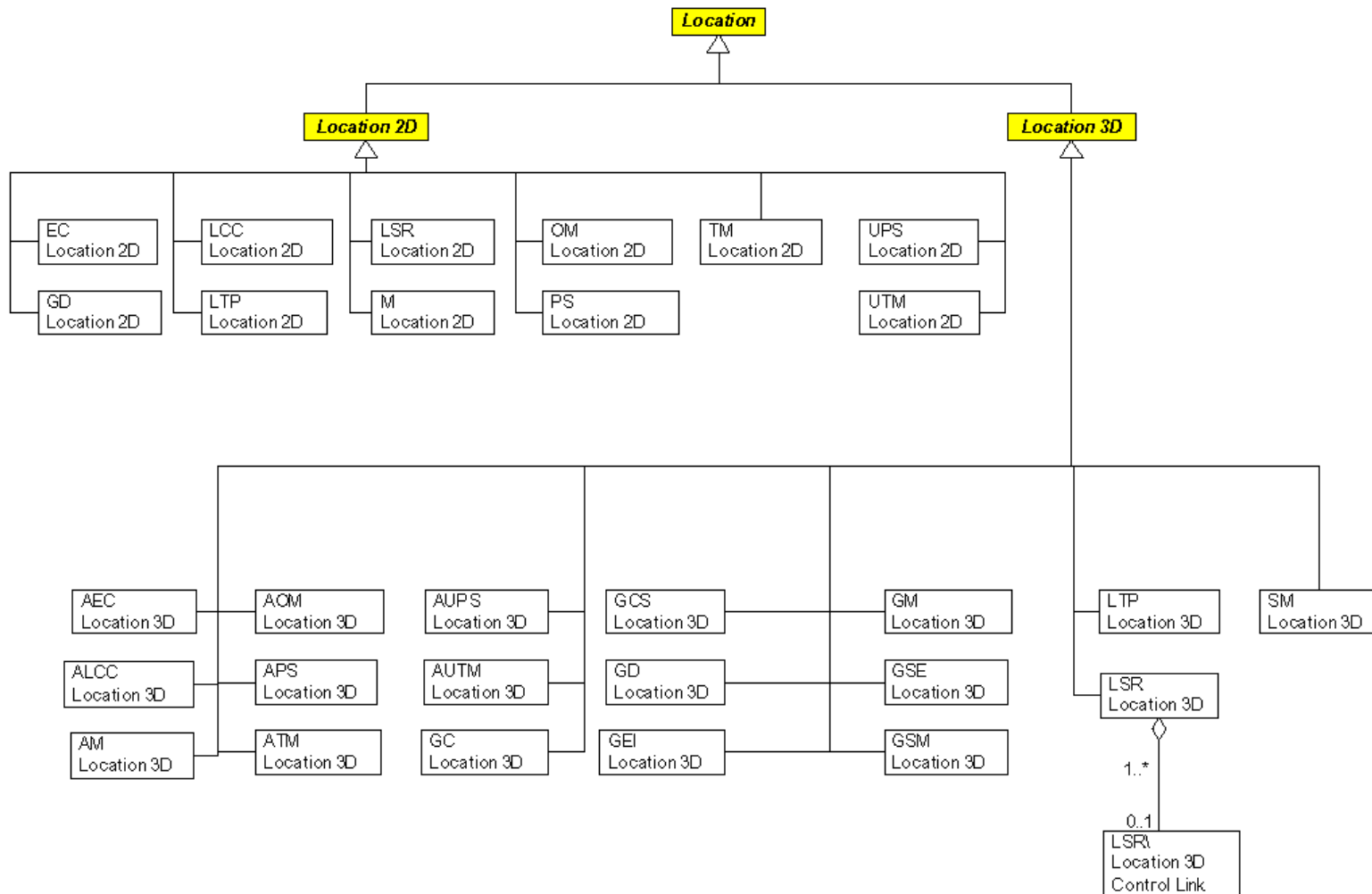


Summary of Location Class Changes

- ◆ **Three abstract classes instead of two**
 - 3.1: 2D/3D
 - 4.0: 2D/Surface/3D classes
- ◆ **One concrete class per SRF Template**
- ◆ **Renamed Geo → Celestio**
- ◆ **Introduced new classes**
- ◆ **Removed:**
 - AUPS 3D/UPS 2D → PS 3D/PS Surface
 - GCS 3D → LTE 3D
 - AUTM 3D/UTM 2D → TM Augmented 3D/TM Surface



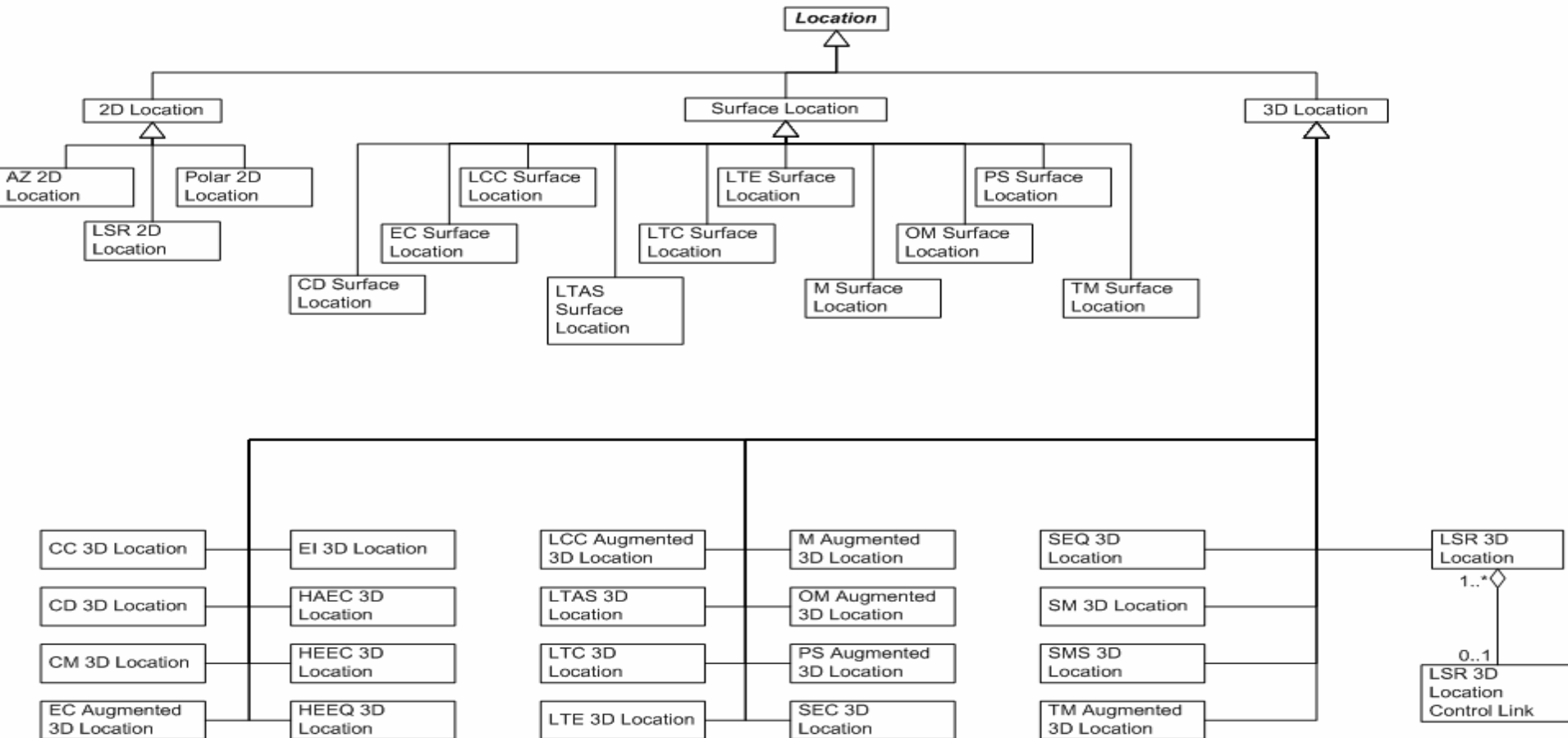
SRM Changes: 3.1 Location Classes





SRM Changes: 4.0 Location Classes

Sheet: 15 Location
Version: 4.0 (10 October 2003)





```

graph TD
    Location[Location] --> 2D[2D Location]
    Location --> Surface[Surface Location]
    Location --> 3D[3D Location]

    2D --> AZ[AZ 2D Location]
    2D --> Polar[Polar 2D Location]
    2D --> LSR_2D[LSR 2D Location]

    Surface --> CD[CD Surface Location]
    Surface --> EC[EC Surface Location]
    Surface --> LCC[LCC Surface Location]
    Surface --> LTAS[LTAS Surface Location]
    Surface --> LTE[LTE Surface Location]
    Surface --> LTC[LTC Surface Location]
    Surface --> M[M Surface Location]
    Surface --> OM[OM Surface Location]
    Surface --> PS[PS Surface Location]
    Surface --> TM[TM Surface Location]

    3D --> CC[CC 3D Location]
    3D --> CD_3D[CD 3D Location]
    3D --> CM[CM 3D Location]
    3D --> EI[EI 3D Location]
    3D --> HAEC[HAEC 3D Location]
    3D --> HEEC[HEEC 3D Location]
    3D --> HEEQ[HEEQ 3D Location]
    3D --> EC_Aug[EC Augmented 3D Location]
    3D --> LCC_Aug[LCC Augmented 3D Location]
    3D --> LTAS_3D[LTAS 3D Location]
    3D --> LTC_3D[LTC 3D Location]
    3D --> LTE_3D[LTE 3D Location]
    3D --> M_Aug[M Augmented 3D Location]
    3D --> OM_Aug[OM Augmented 3D Location]
    3D --> PS_Aug[PS Augmented 3D Location]
    3D --> SEC[SEC 3D Location]
    3D --> SEQ[SEQ 3D Location]
    3D --> SM[SM 3D Location]
    3D --> SMS[SMS 3D Location]
    3D --> TM_Aug[TM Augmented 3D Location]
    3D --> LSR_3D[LSR 3D Location]

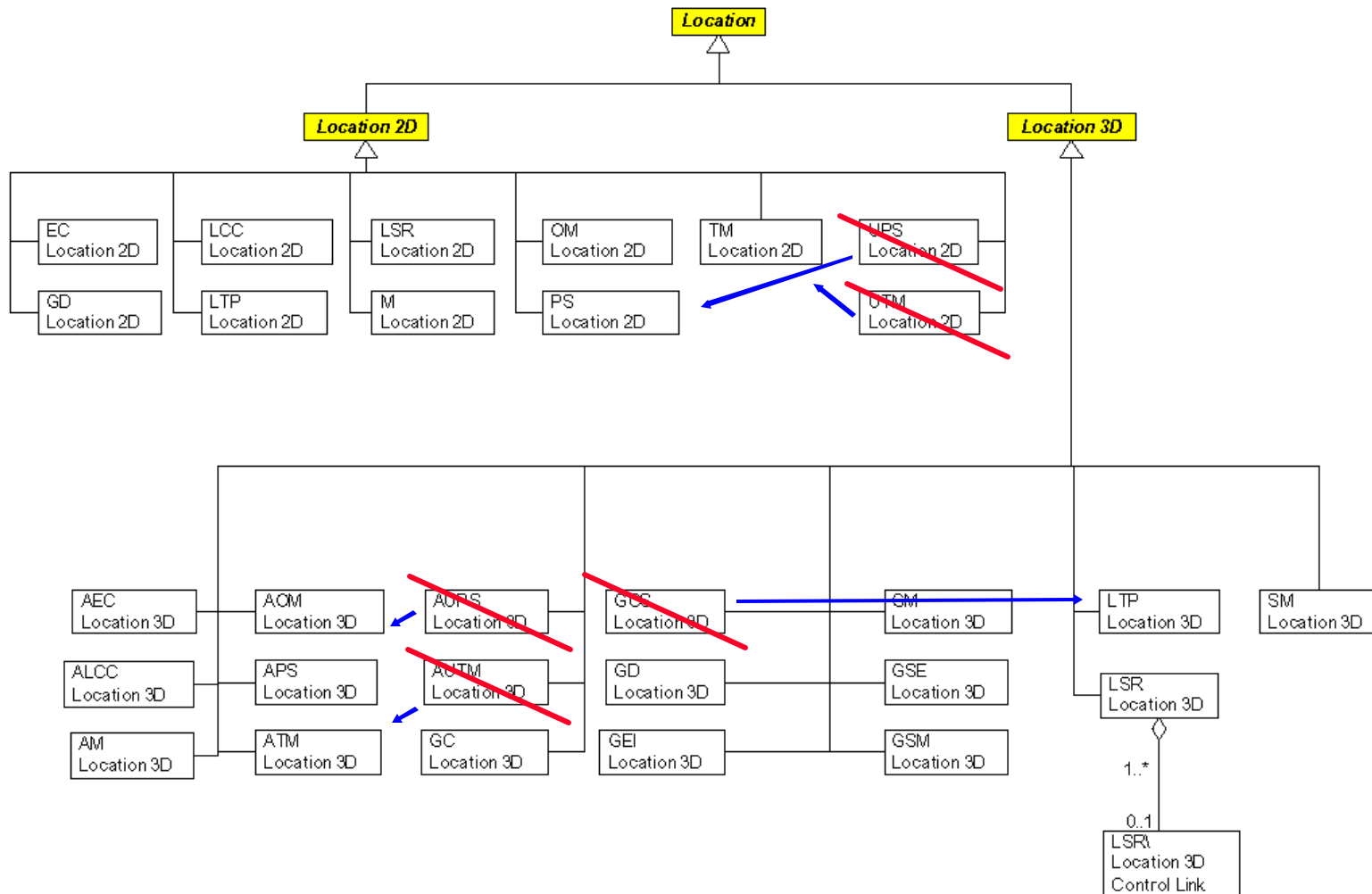
    LSR_3D -- "1..*" --> LSR_Link[LSR 3D Location Control Link]
    LSR_Link -- "0..1" --> LSR_3D

    subgraph Dashed_Box [ ]
        LTAS_Surf[LTAS Surface Location]
        LTC_Surf[LTC Surface Location]
        EI_3D[EI 3D Location]
        HAEC_3D[HAEC 3D Location]
        HEEC_3D[HEEC 3D Location]
        HEEQ_3D[HEEQ 3D Location]
        LCC_Aug_3D[LCC Augmented 3D Location]
        LTAS_3D[LTAS 3D Location]
        LTC_3D[LTC 3D Location]
        LTE_3D[LTE 3D Location]
    end

```



SRM Changes: Deleted Location Classes





SRM Units in Transmittals

- ◆ In 3.1, 1st and 2nd value of location data were given in metres and 3rd value was either metres or feet for linear data.
- ◆ In 3.1 location values of angular data were provided in arcdegrees.
- ◆ In 4.0, the units for location data are specified in the **SE_SRF_Info** fields:
EDCS_Unit_Code angular_unit;
EDCS_Unit_Code linear_unit;
EDCS_Scale_Code linear_scale;
- ◆ The units specified in the **SE_SRF_Info** applies to all location data within scope. Thus all linear location values must be consistent.



SRF Parameters in a Transmittal

- ◆ **In 3.1, there was only one way to store the SRF parameters information.**
 - Large structure with one entry per SRF parameters
- ◆ **For 4.0, all 3.1 SRF parameters map to a smaller set of 4.0 Template parameters, so information is condensed**
 - Example, UTM parameters map to TM parameters
- ◆ **In 4.0, Set Code and Set Member Codes are provided with defined parameter information such as UTM & GCS**
- ◆ **In 4.0, SRF Codes are defined for specific SRFs**
 - SRF Codes are also provided with defined parameter information



SRF Parameters in a Transmittal

◆ SRF information can be stored in three different manners:

1. *Using Set/Set Member*
2. *SRF Instance codes*
3. *Using SRF Template parameters*

◆ Example of 2 exclusive methods for specifying the UTM Southern Hemisphere Zone 12 SRF information:

- Using Set/Set Member:
 - Set Code: **SRFS_UNIV_TRANS_MERC;**
 - Member Code: **SRM_UTM_SOUTHERN_HEMI_12**
- Using TM SRF Template parameters:
orm = SRM_ORM_WGS_1984;
origin_longitude = 0.0
origin_latitude = -111.0
central_scale = 0.9996
false_easting = 500000.0
false_northing = 0.0



Modified SRM structures

◆ **SRM_SRF_Parameters → SE_SRF_Info**

- Vertical Datum separated from SRF parameters and corresponds to Vertical Offset Surface Code
- Unit information made explicit

typedef struct

{

SRM_Vertical_Offset_Surface_Code vos_code;

EDCS_Unit_Code angular_unit;

EDCS_Unit_Code linear_unit;

EDCS_Scale_Code linear_scale;

SRM_SRF_Parameters_Info srf_params_info

} SE_SRF_Info;



SRF Parameters Info

```
typedef struct
{
    SRM_SRF_Params_Info_Code srf_params_info_code;
    union
    {
        SRM_SRF_Template_Parameters    srf_template;
        SRM_SRF_Set_Info                srf_set;
        SRM_SRF_Code                   srf_instance;
    } srf_params_info;
} SRM_SRF_Parameters_Info;
```



SRF Template Parameters

```
typedef struct
{
    SRM_SRF_Template_Code srf_template;
    union
    {
        SRM_ORM_Parameters          cc_srf_params;
        SRM_LSR_3D_SRF_Parameters    lsr_3d_srf_params;
        SRM_LSR_2D_SRF_Parameters    lsr_2d_srf_params;
        SRM_ORM_Parameters          cd_srf_params;
        . . .
        SRM_EC_SRF_Parameters        ec_srf_params;
    } parameters;
} SRM_SRF_Template_Parameters;
```



ORM Parameters

◆ ORM_Code to ORM_Code, HSR_Code

- Problem because need to be able to differentiate between specification and transformation
- ORM is the reference model
- HSR is the transformations applied to this model
 - *Allows no transformation as well*

```
typedef struct
{
    SRM_ORM_Code    orm_code;
    SRM_HSR_Code    hsr_code;
} SRM_ORM_Parameters
```



SRF Parameters

```
typedef struct
{
    SRM_Long_Float    origin_longitude;
    SRM_Long_Float    origin_latitude;
    SRM_Long_Float    central_scale;
    SRM_Long_Float    false_easting;
    SRM_Long_Float    false_northing;
} SRM_TM_Parameters;
```

```
typedef struct
{
    SRM_ORM_Parameters    orm_params;
    SRM_TM_Parameters     tm_params;
} SRM_TM_SRF_Parameters;
```



SRF Set Info

```
typedef struct
{
    SRM_SRF_Params_Info_Code srf_params_info_code;
    union
    {
        SRM_SRF_Template_Parameters    srf_template;
        SRM_SRF_Set_Info               srf_set;
        SRM_SRF_Code                   srf_instance;
    } srf_params_info;
} SRM_SRF_Parameters_Info;
```

```
typedef struct
{
    SRM_SRF_Set_Code               set_code;
    SRM_SRF_Set_Member_Code        member_code;
    SRM_ORM_Parameters              orm_params
} SRM_SRF_Set_Info;
```



SRF Parameters in a Transmittal: Storing in a Transmittal

//Storing SRF Set/Set Member Codes

srf_param_info_code = SRM_SRFPI_SET;

srf_param_info.srf_set.set_code = SRM_SRFS_UNIV_TRANS_MERC;

srf_param_info.set.set.member_code = SRM_UTM_SOUTHERN_HEMI_12;

srf_param_info.set.orm_params.orm_code = SRM ORM_WGS_1984;

srf_param_info.set.orm_params.hsr_code = SRM_HSR_WGS_1984;



SRF Parameters in a Transmittal: Storing in a Transmittal

```
// TM Template Parameters  
srf_param_info_code = SRF_INFO_CODE_TEMPLATE;  
srf_param_info.parameters.srf_template = SRFT_TRANSVERSE_MERCATOR;  
srf_param_info.parameters.tm_srf_params.orm_params.orm_code =  
SRM ORM_WGS_1984;  
srf_param_info.parameters.tm_srf_params.orm_params.hsr_code =  
SRM_HSR_WGS_1984;  
srf_param_info.parameters.tm_srf_params.tm_params.origin_longitude = 0.0  
srf_param_info.parameters.tm_srf_params.tm_params.origin_latitude = -  
111.0  
srf_param_info.parameters.tm_srf_params.tm_params.central_scale = 0.9996  
srf_param_info.parameters.tm_srf_params.tm_params.false_easting =  
500000.0  
srf_param_info.parameters.tm_srf_params.tm_params.false_northing = 0.0
```