Microsoft Software Annotation Language

SAL: Introduction

MSDN: "*The Microsoft source-code annotation language (SAL) provides a set of annotations that you can use to describe how a function uses its parameters, the assumptions that it makes about them, and the guarantees that it makes when it finishes.*"

SAL: Content

1) Parameters
2) Return Values
3) Function Behaviour
4) Structs and classes
5) *Locking
6) *Conditional annotations

SAL: Parameters

```
void * memcpy(
    void *dest,
    const void *src,
    size_t count
);
```

SAL: Parameters

```
void * memcpy(
    void *dest,
    const void *src,
    size_t count
)
{
    int i;

    for (i = 0; i < count; i++)
    {
        ((BYTE*)dest)[i] = ((BYTE*)src)[i];
    }
}
```

SAL: Parameters

```
void * memcpy(
    _Out_ void *dest,
    _In_ const void *src,
    size_t count
)
{
    int i;

    for (i = 0; i < count; i++)
    {
        ((BYTE*)dest)[i] = ((BYTE*)src)[i];
    }
}
```

SAL: Parameters

```
void * memcpy(
    _Out_writes_bytes_all_(count) void *dest,
    _In_reads_bytes_(count) const void *src,
    size_t count
)
{
    int i;

    for (i = 0; i < count; i++)
    {
        ((BYTE*)dest)[i] = ((BYTE*)src)[i];
    }
}
```

SAL: Retrun value

```
_Success_(return != NULL)
_Must_inspect_result_
void * memcpy(
    _Out_writes_bytes_all_(count) void *dest,
    _In_reads_bytes_(count) const void *src,
    size_t count
)
{
    int i;

    for (i = 0; i < count; i++)
    {
        ((BYTE*)dest)[i] = ((BYTE*)src)[i];
    }
}
```

SAL: Function behaviour

```
_Maybe_raises_SEH_exception_

_IRQL_requires_max_(APC_LEVEL)

NTKERNELAPI

VOID

NTAPI

ProbeForRead (

    __in_data_source(USER_MODE) _In_reads_bytes_(Length)
volatile VOID *Address,

    _In_ SIZE_T Length,

    _In_ ULONG Alignment

    );
```

SAL: Structs and classes

```
//
// USER_DATA - our user data
//
typedef struct _USER_DATA
{
    _Field_size_(FIELD_SIZE) CHAR Username[FIELD_SIZE];
    _Field_size_(FIELD_SIZE) CHAR Name[FIELD_SIZE];
    _Field_size_(FIELD_SIZE) CHAR Email[FIELD_SIZE];
    _Field_size_(FIELD_SIZE) CHAR Password[FIELD_SIZE];
} USER_DATA, *PUSER_DATA;
```

SAL: Structs and classes

```
//
// CREATE_PROCESS_INFORMATION
//
_Struct_size_bytes_(sizeof(struct _CREATE_PROCESS_INFORMATION) \
+ ProcessPathSize + ParentPathSize)
typedef struct _CREATE_PROCESS_INFORMATION
{
    DWORD Pid;
    DWORD ParentPid;
    DWORD ProcessPathSize;
    DWORD ParentPathSize;
    BYTE  Buffer[0];
}CREATE_PROCESS_INFORMATION, *PCREATE_PROCESS_INFORMATION;
```

SAL: Locking

```
WINBASEAPI
_Acquires_exclusive_lock_(*SRWLock)
VOID
WINAPI
AcquireSRWLockExclusive(
    _Inout_ PSRWLOCK SRWLock
    );
```

```
WINBASEAPI
_Acquires_shared_lock_(*SRWLock)
VOID
WINAPI
AcquireSRWLockShared(
    _Inout_ PSRWLOCK SRWLock
    );
```

SAL: Locking

```
WINBASEAPI
_Releases_exclusive_lock_(*SRWLock)
VOID
WINAPI
ReleaseSRWLockExclusive(
    _Inout_ PSRWLOCK SRWLock
    );
```

```
WINBASEAPI
_Releases_shared_lock_(*SRWLock)
VOID
WINAPI
ReleaseSRWLockShared(
    _Inout_ PSRWLOCK SRWLock
    );
```

SAL: Conditional annotations

```
WINBASEAPI
_When_(return!=0, _Acquires_exclusive_lock_(*SRWLock))
BOOLEAN
WINAPI
TryAcquireSRWLockExclusive(
    _Inout_ PSRWLOCK SRWLock
    );
```

SAL: Conditional annotations

```
WINUSERAPI

_Success_(return)

int

WINAPI

DrawTextA(

    _In_ HDC hdc,

    _When_((format & DT_MODIFYSTRING), _At_((LPSTR)lpchText,
_Inout_grows_updates_bypassable_or_z_(cchText, 4)))

    _When_((!(format & DT_MODIFYSTRING)),
_In_bypassable_reads_or_z_(cchText))

    LPCSTR lpchText,

    _In_ int cchText,

    _Inout_ LPRECT lprc,

    _In_ UINT format);
```

SAL: Conditional annotations

```
#define _In_bypassable_reads_or_z_(size)                              \
    _When_(((size) == -1) || (_String_length_(_Curr_) <               \
(size)), _In_z_) \                                                    \
    _When_(((size) != -1) && (_String_length_(_Curr_) >=              \
(size)), _In_reads_(size))                                            \


#define _Inout_grows_updates_bypassable_or_z_(size, grows) \
    _When_(((size) == -1) || (_String_length_(_Curr_) <           \
(size)), _Pre_z_ _Pre_valid_                                      \
_Out_writes_z_(_String_length_(_Curr_) + (grows)))                \
    _When_(((size) != -1) && (_String_length_(_Curr_) >=          \
(size)), _Pre_count_(size) _Pre_valid_ _Out_writes_z_((size) + \
 (grows)))
```