## 1. ****Protocolos Nativos en NS-3****

**AODV, OLSR, DSDV, DSR**  
Estos están incluidos en NS-3 y se habilitan usando sus respectivos helpers en el script de simulación, por ejemplo:

cpp

InternetStackHelper stack;

AodvHelper aodv;

stack.SetRoutingHelper(aodv);

stack.Install(nodes);

*// Para OLSR, DSDV, DSR, cambia 'aodv' por 'olsr', 'dsdv', 'dsr'*

## 2. ****HWMP (IEEE 802.11s Mesh)****

HWMP está integrado en el módulo mesh de NS-3. Ejemplo básico para usar HWMP:

cpp

MeshHelper mesh;

mesh.SetStackInstaller("ns3::Dot11sStack");

mesh.SetRoutingProtocol("HWMP");

mesh.Install(wifi, nodes);

## 3. ****Babel****

Babel no viene por defecto, pero existen módulos externos para NS-3. Para integrarlo:

* Descarga el módulo Babel compatible para NS-3 (por ejemplo, desde un repositorio como GitHub).
* Copia el código fuente del módulo a src/babel dentro de tu NS-3.
* Modifica el archivo wscript para incluir el nuevo módulo en la compilación.
* Compila NS-3 nuevamente.

En tu script, utiliza el helper de Babel (según la documentación del módulo externo):

cpp

BabelHelper babel;

InternetStackHelper stack;

stack.SetRoutingHelper(babel);

stack.Install(nodes);

(Debes consultar la documentación del módulo Babel específico para detalles exactos de instalación y uso)

Existe una **implementación de Babel para NS-3** desarrollada por la Universidad Técnica de Múnich (TUM), documentada en el paper "An Implementation of the Babel Routing Protocol for ns3" (2022)

. Este módulo es de código abierto y se diseñó específicamente para facilitar la simulación de Babel en NS-3. A continuación te explico cómo descargarlo e integrarlo:

## 1. ****Dónde encontrar el módulo****

* El módulo y sus fuentes están descritos en el paper citado
* . El paper incluye detalles técnicos y de integración, y suele estar acompañado de un repositorio público (consulta el paper para el enlace directo; si no aparece, puedes contactar a los autores, cuyas direcciones de correo figuran en el documento).

## 2. ****Características del módulo Babel para NS-3****

* Implementa Babel como protocolo de enrutamiento IPv6 (no IPv4).
* Proporciona un helper para instalar Babel en nodos NS-3.
* Permite ajustar parámetros como los intervalos de Hello, IHU y Update TLVs mediante atributos NS-3.
* Permite excluir interfaces de Babel, útil para escenarios híbridos.
* Incluye ejemplos y tests básicos para validar la instalación.

## 3. ****Cómo descargar e integrar Babel en NS-3****

## a) ****Descarga****

* Revisa el paper para el enlace al repositorio (usualmente en GitHub o GitLab). Si no está disponible, puedes buscar en Google "babel ns-3 github" o contactar a los autores ([malte.von.ehren@tum.de](mailto:malte.von.ehren@tum.de), [andre@net.in.tum.de](mailto:andre@net.in.tum.de), [wiedner@net.in.tum.de](mailto:wiedner@net.in.tum.de))
* .
* Alternativamente, revisa proyectos como [meshnet-lab](https://github.com/mwarning/meshnet-lab) que incluyen experimentos y scripts para Babel, aunque suelen estar más orientados a pruebas de sistemas reales que a simulación NS-3
* .

## b) ****Integración en NS-3****

1. **Copia el módulo**  
   Coloca la carpeta del módulo (normalmente llamada babel) dentro de src/ en tu árbol de NS-3.
2. **Edita el archivo** wscript  
   Añade el módulo Babel a la lista de módulos a compilar en el archivo raíz wscript de NS-3.
3. **Compila NS-3**  
   Ejecuta:

* bash

./waf configure

./waf build

* **Usa el helper en tu script**  
  En tu código de simulación, instala Babel así:

1. cpp

BabelHelper babel;

InternetStackHelper stack;

stack.SetRoutingHelper(babel);

stack.Install(nodes);

Puedes ajustar los parámetros usando los atributos del helper, por ejemplo para modificar los intervalos de Hello o Update.

## 4. ****Limitaciones****

* **Solo IPv6**: La implementación es para IPv6, no para IPv4
* .
* **No todas las funciones del RFC**: Algunas características avanzadas de Babel pueden no estar implementadas, pero para simulaciones estándar es funcional.
* **No interoperable fuera de NS-3**: Solo funciona entre nodos NS-3 con la misma implementación.

## 5. ****Referencias****

"An Implementation of the Babel Routing Protocol for ns3", Malte von Ehren et al., TUM, 2022 ([PDF]

* )
* meshnet-lab/README.md at master - GitHub

**En resumen:**  
Puedes descargar el módulo Babel para NS-3 desde el repositorio indicado en el paper de TUM o contactando a los autores. La integración sigue el patrón estándar de módulos externos en NS-3: copia a src/, edita wscript, recompila y usa el helper en tus scripts. La implementación es funcional para simulaciones académicas y comparativas en entornos IPv6.

## 4. ****BATMAN****

BATMAN **no está incluido** oficialmente en NS-3 ni existe un módulo mantenido en los repositorios principales. Para agregarlo:

* **Opción 1:** Busca un módulo BATMAN para NS-3 en repositorios de la comunidad, papers o foros. Si existe, sigue el proceso similar al de Babel: descarga, copia a src/batman, modifica wscript, compila y usa el helper correspondiente.
* **Opción 2:** Si no hay módulo disponible, deberías implementarlo tú mismo siguiendo el patrón de otros protocolos en NS-3:
  + Crea un nuevo módulo en src/batman con subdirectorios model, helper, y examples.
  + Implementa la lógica de BATMAN como una clase que herede de Ipv4RoutingProtocol.
  + Crea el helper para integrarlo al stack.
  + Modifica wscript y recompila

Sí existe un módulo BATMAN para NS-3, aunque **no es oficial ni está en los repositorios principales**, pero está disponible como parte de proyectos de código abierto y ha sido utilizado en investigaciones académicas.

## Resumen de la Información Encontrada

## 1. ****Módulo BATMAN para NS-3: BATSEN****

* El repositorio [BATSEN](https://github.com/npowell3/BATSEN) incluye un módulo de BATMAN para NS-3, junto con otros módulos para redes de sensores y estadísticas
* .
* El archivo batmand-routing-protocol.cc implementa la lógica central del protocolo BATMAN en NS-3
* .
* Este módulo fue diseñado para experimentos con redes inalámbricas y sensores, y es de código abierto.

## 2. ****Uso en Investigación y Comparaciones****

* BATMAN ha sido utilizado en simulaciones académicas para comparar su desempeño con HWMP en NS-3, evaluando métricas como PDR, throughput y delay en distintos tamaños de red
* .
* El protocolo implementado en NS-3 sigue el funcionamiento de BATMAN capa 3, utilizando OGMs (Originator Messages) y métricas como RQ, EQ y TQ para la elección de rutas
* .

## 3. ****Desarrollo y Estado de la Comunidad****

* Hubo un esfuerzo comunitario para desarrollar un módulo BATMAN para NS-3, integrándolo como plugin del módulo mesh, aunque el desarrollo se pausó en 2020
* .
* No está incluido en las versiones oficiales de NS-3, pero el código fuente del módulo está disponible y puede ser adaptado a versiones recientes de NS-3 con algunos ajustes
* .

## Cómo Integrar BATMAN en NS-3

1. **Descarga el módulo** desde el repositorio BATSEN:
   * <https://github.com/npowell3/BATSEN>
2. **Copia el código** del módulo BATMAN (carpeta batmand) a la carpeta src/ de tu NS-3.
3. **Modifica el archivo** wscript de NS-3 para incluir el módulo en la compilación.
4. **Compila NS-3** nuevamente.
5. **Utiliza el helper** y las clases del módulo BATMAN en tus scripts de simulación, siguiendo la estructura de los ejemplos incluidos en el repositorio.

En la literatura, BATMAN implementado en NS-3 ha demostrado buen desempeño en redes mesh densas, superando a HWMP en ciertas métricas como el PDR en escenarios de alta densidad nodal

.

## Referencias directas

* [Repositorio BATSEN con BATMAN para NS-3]
* [Implementación central del protocolo en NS-3]
* [Comparaciones académicas BATMAN vs HWMP en NS-3]
* [Desarrollo comunitario y discusión técnica]

**En resumen:**  
Puedes integrar BATMAN en NS-3 usando el módulo del proyecto BATSEN, aunque requiere pasos manuales de integración y adaptación. Es funcional para simulaciones académicas y comparativas, pero no es parte del core oficial de NS-3

.